Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи нейромережевої ідентифікації нелінійних динамічних об’єктів

(тема)

Виконав:
студент II курсу, групи СПм-18-3
Щербаков П.Ю.
(прізвище, ініціали)

Спеціальність
123 – Комп’ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: проф. Руденко О.Г.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ ______ Коваленко А.А.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп’ютерна інженерія
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри ____________
(підпис)
“____”________________ 20__ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Щербакову Павлу Юрійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Методи нейромережевої ідентифікації нелінійних динамічних об’єктів

затверджена наказом по університету від “ 30 ” березня 2020 р. № 478 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 18 травня 2020 р.

3. Вхідні дані до роботи

4. Перелік питань, що потрібно опрацювати в роботі

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп’ютерних ілюстрацій (слайдів) ______________________________

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Огляд існуючих методів ідентифікації	31.03.20-13.04.20	
	нелінійних динамічних об’єктів		
2	Вибір та обґрунтування методики дослідження	14.04.20-21.04.20	
3	Вибір інструментальних засобів	22.04.20-27.04.20	
4	Проведення імітаційного моделювання	28.04.20-05.05.20	
5	Оформлення матеріалів атестаційної роботи	06.05.20-11.05.20	
6	Подання атестаційної роботи керівникові	12.05.20-13.05.20	
	та її попередній захист		
7	Подання атестаційної роботи на рецензування	14.05.20-15.05.20	

Дата видачі завдання 30 березня 2020 р.

Студент ____________________
(підпис)

Керівник роботи ____________________ проф. Руденко О.Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 82 с., 29 рис., 3 табл., 1 дод., 33 джерел.

ШТУЧНА НЕЙРОННА МЕРЕЖА, РАДІАЛЬНО-БАЗИСНА МЕРЕЖА, НЕЛІНІЙНИЙ ДИНАМІЧНИЙ ОБ’ЕКТ, МЕТОД ЛЕВЕНБЕРГА-МАРКВАРДТА, ІДЕНТИФІКАЦІЯ.

Метою атестаційної роботи є побудова моделі ідентифікації нелінійних динамічних об’єктів на основі радіально-базисних мереж.

У ході виконання атестаційної роботи були досліджені існуючі методи ідентифікації нелінійних динамічних об’єктів. На підставі проведеної роботи була побудована модель ідентифікації нелінійних динамічних об’єктів на основі радіально-базисної мережі.

Проведене імітаційне моделювання наступних ряду алгоритмів навчання радіально-базисної мережі та представлена їх порівняльна характеристика.

Показана ефективність використання алгоритму Левенберга-Марквардта при моделюванні нелінійних динамічних об’єктів без перешкод, а також при їх присутності.

ABSTRACT

Master’s thesis: 82 pages, 29 figures, 3 tables, 1 appendices, 33 sources.

ARTIFICIAL NEURAL NETWORK, RADIAL BASIS FUNCTION NETWORK, NONLINEAR DYNAMIC OBJECT, LEVENBERG-MARKWARDT METHOD, IDENTIFICATION.

The major goal of this thesis is to build a model for the identification of nonlinear dynamic objects based on radial-basis networks.

Existing identification methods of nonlinear dynamic objects are investigated in this thesis. A model for the identification of nonlinear dynamic objects based on a radial-basis network have been developed.

The simulation modeling of the following number of algorithms of training of a radial-basis network is carried out and their comparative characteristic is presented.

The efficiency of using the Levenberg-Marquardt algorithm in nonlinear dynamic objects without obstacles modeling as well as in their presence is shown.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БП – багатошаровий персептрон

НДО – нелінійний динамічний об’єкт

РБМ – радіально-базисна мережа

ШНМ – штучна нейронна мережа

CMAC (Cerebellar Model Articulation Controller) – церебральна модель артикуляції контролера

ВСТУП

Проблема управління нелінійними динамічними системами відноситься до розряду найважливіших завдань теорії автоматичного управління. Поряд з методами синтезу керуючих пристроїв і систем сучасна теорія управління включає методи ідентифікації та оцінювання станів процесів. Різні алгоритми ідентифікації процесів дозволяють визначити структуру моделі об’єкта управління і відновити параметри цієї моделі як безпосередньо в контурі управління, так і поза ним. Ці алгоритми використовуються при синтезі не тільки звичайних, але і адаптивних систем управління.

Основою класичної теорії управління є добре розвинена теорія лінійних систем. Найчастіше елементи цієї теорії використовуються при дослідженні і синтезі систем управління нелінійними об’єктами, однак, хороші результати можна отримати лише в тих випадках, коли нелінійність об’єкта є несуттєвою. Найчастіше застосовується підхід, заснований на апроксимації нелінійностей рядами Вольтерра, Гаммерштейна, Вінера, тощо.

Однак, застосування даних моделей обмежене, так як їх використання передбачає попереднє рішення задачі структурної ідентифікації для визначення порядку моделі. Наявність зовнішніх збурень і похибок вимірювань вимагає додаткової обробки і істотно ускладнює рішення задачі управління, а необхідність управління динамічними об’єктами в реальному масштабі часу ставить перед розробником завдання створення швидкодіючих алгоритмів ідентифікації.

Наявність зовнішніх збурень і помилок вимірювань, не дозволяє створити єдиний підхід при побудові систем ідентифікації. При вирішенні практичних завдань класична модель може не забезпечити необхідний критерій якості ідентифікації, так як вона, побудована на припущенні про лінійність системи та може не відображати її дійсних властивостей.

Останнім часом досить багатообіцяючою альтернативою класичним методам побудови систем ідентифікації нелінійних об’єктів є штучні нейронні мережі. Завдяки наявності в структурі ШНМ нейронів з нелінійними функціями активації, нейромережа здатна апроксимувати істотно нелінійні функції з будь-яким ступенем точності. Той факт, що параметри мережі налаштовуються лише на основі даних про вхід і вихід, дає можливість використання нейромереж в умовах невизначеності.

Метою атестаційної роботи є побудова моделі ідентифікації нелінійних динамічних об’єктів на основі радіально-базисних мереж. Досягнення поставленої мети здійснюється вирішенням наступних основних завдань:

- аналіз існуючих методів опису та ідентифікації нелінійних динамічних об’єктів;
- аналіз принципів побудови ШНМ;
- побудова моделі нейромережевої ідентифікації НДО;
- дослідження швидкості збіжності алгоритмів навчання ШНМ;
- проведення імітаційне моделювання ідентифікації на основі РБМ.

Об’єктом дослідження є нелінійні динамічні об’єкти, що функціонують в умовах апріорної і поточної невизначеності про властивості об’єктів і діючих на них перешкод.

Предметом дослідження є нейромережеве моделювання і схеми ідентифікації, побудовані на основі РБМ.

1 ОГЛЯД ОПИСУ НЕЛІНІЙНИХ ОБ’ЄКТІВ

1.1 Математичні методи опису нелінійних об’єктів

У загальному вигляді задача побудови математичної моделі об’єкта полягає у виборі її структури та оцінки її параметрів таким чином, щоб при використанні критерію мінімуму деякої функції різниці розрахункових і експериментальних даних дотримувалася умова близькості моделі досліджуваного процесу [1,2].

Вибір конкретного типу моделі пов’язаний з аналізом апріорної інформації. Апріорне знання об’єкта управління базується на загальному його розумінні, фізичних законах, попередній обробці інформації. На цьому етапі формуються міркування про лінійність або нелінійність, стаціонарність або нестаціонарність, структуру, вибір вхідного сигналу. В результаті формуються системи рівнянь, що описують внутрішній стан системи і зовнішні сигнали[2].

1.1.1 Опис статичних нелінійних систем

Статична нелінійна система, на вхід якої подається сигнал $u(t)$, а на виході формується сигнал $y(t)$ може бути представлена непараметричною моделлю у вигляді ряду Тейлора:

$$y(t)=c_0+c_1u(t)+c_2u^2(t)+...+=\sum_{i=0}^{N}c_iu^i(t)\,. \qquad (1.1)$$

Точність цієї моделі тим вище, чим більше використовується членів ряду, хоча зазвичай обмежуються розкладанням до членів другого порядку. Такі моделі використовуються при визначенні стаціонарних режимів функціонування об’єктів і їх підтримці [3].

Також широко використовуються параметричні моделі, засновані на поліноміальній апроксимації. Так поліноміальна модель для p входів порядку l описується рівнянням:

$$y(t)=c_0+\sum_{i=1}^{p} c_i u_i(t)+\sum_{i_1=1}^{p}\sum_{i_2=i_1}^{p} c_{i_1 i_2} u_{i_1}(t) u_{i_2}(t)+\ldots+ \\ +\sum_{i_1=1}^{p}\ldots\sum_{i_l=i_{l-1}}^{p} c_{i_1\ldots i_l} u_{i_1}(t)\ldots u_{i_l}(t). \qquad (1.2)$$

Моделі (1.1) та (1.2) враховують тільки статичні властивості об’єкта, однак, більшість же реальних об’єктів є динамічними, що в свою чергу вимагає відображення їх динамічних властивостей в моделях.

1.1.2 Модель Вольтерра

Для опису безперервних нелінійних динамічних систем спочатку використовувався інтеграл Вольтерра:

$$y(t)=g_0+\int_{\tau_1=0}^{t} g_1(\tau_1)u(t-\tau_1)d\tau_1+\int_{\tau_1=0}^{t}\int_{\tau_2=0}^{t} g_2(\tau_1,\tau_2)u(t-\tau_1)u(t-\tau_2)d\tau_1 d\tau_2+\ldots= \\ =\sum_{i=0}^{n}\int_{\tau_1=0}^{t}\ldots\int_{\tau_i=0}^{t} g_i(\tau_1,\ldots,\tau_i)\prod_{j=1}^{i}u(t-\tau_j)d\tau_1\ldots d\tau_i\ ), \qquad (1.3)$$

де $g_n(\tau_1,\ldots\tau_n)$, $n=0,1,2,\ldots$ – ядра Вольтерра n-го порядку для безперервної моделі.

А для дискретних нелінійних динамічних систем – ряд Вольтерра:

$$y(k)=h_0+\sum_{\kappa_1=0}^{k}h_1(\kappa_1)u(k-\kappa_1)+\sum_{\kappa_1=0}^{k}\sum_{\kappa_2=0}^{k}h_2(\kappa_1,\kappa_2)u(k-\kappa_1)u(k-\kappa_2)+...=$$
$$=\sum_{i=0}^{n}...\sum_{\kappa_i=0}^{k}h_i(\kappa_1,...,\kappa_i)\prod_{j=1}^{i}u(k-\kappa_j), \quad (1.4)$$

де $g_n(\tau_1,...\tau_n)$, $n=0,1,2,...$ – ядра Вольтерра n-го порядку для дискретної моделі.

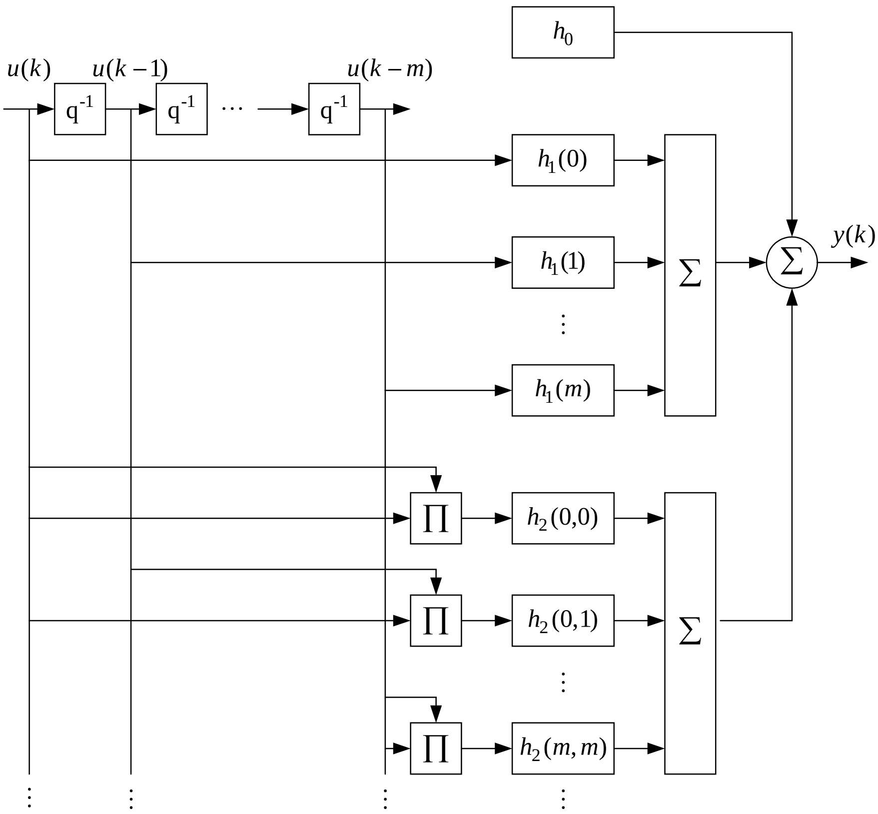

Рисунок 1.1 – Дискретна модель Вольтерра другого порядку

Важливим завданням є вибір кількості членів ряду Вольтерра, що адекватно описують досліджуваний реальний об’єкт. Визначення порядку моделі відноситься до задачі структурної ідентифікації. У зв’язку з тим, що в даний час немає загальних рекомендацій щодо її вирішення, то при практичному застосуванні зазвичай обмежуються нелінійністю другого порядку, нехтуючи членами розкладання (1.3), (1.4) третього і більш високих порядків. На рисунку 1.1 представлена квадратична модель Вольтерра. Істотним недоліком моделі є велика кількість параметрів, що входять в її опис. Навіть в тих випадках, коли система є лінійною, необхідно використовувати від 10 до 20 точок для апроксимації вагової функції [4].

1.1.3 Модель Урисона та її модифікації

Найбільш загальною моделлю уявлення довільного динамічного об’єкта є модель Урисона [4]:

$$y(t)=\int_0^t K\left[t,\tau,u(\tau)d\tau\right], \tag{1.4}$$

де K[·] – деякий нелінійний оператор.

Однак використання настільки загальної моделі не представляється можливим. Різні види апроксимації моделі Урисона були запропоновані в роботах [2, 3]. Галлман П.Г. і Нарендра К.С. [4] досліджували модель Урисона за допомогою більш простих моделей – моделей Гаммерштейна. Модель Урисона для безперервного часу має вигляд:

$$\begin{aligned} y(t)=&g_0+\sum_{i_1=0}^{n}\int_{\tau_1=0}^{t} g_{1i_1}(\tau_1)P_{1i_1}[u(t-\tau_1)]d\tau_1+\\ &+\sum_{i_1=0}^{n}\sum_{i_2=0}^{n}\left[\int_{\tau_1=0}^{t} g_{21i_1}(\tau_1)P_{21i_1}[u(t-\tau_1)]d\tau_1\right]\left[\int_{\tau_2=0}^{t} g_{22i_1}(\tau_1)P_{22i_1}[u(t-\tau_2)]d\tau_2\right]+...=\\ &=\sum_{i=0}^{n}\sum_{i_1=0}^{n}...\sum_{i_n=0}^{n}\prod_{j=1}^{i}\int_{\tau_j=0}^{t} g_{iji_j}(\tau_j)P_{iji_j}[u(t-\tau_j)]d\tau_j; \end{aligned} \tag{1.4}$$

Для дискретного часу:

$$y(k)=\sum_{i=0}^{n}\sum_{i_1=0}^{n}...\sum_{i_n=0}^{n}\prod_{j=1}^{i}\sum_{\kappa_n=0}^{n} h_{iji_j}(\kappa_j)P_{iji_j}[u(k-\kappa_j)], \tag{1.5}$$

де $P[u(\times)]=\sum_{i=1}^{n}\gamma_i u^i(\times)$ – нелінійність.

Структура моделі (1.5), обмеженою нелінійністю другого порядку, наведена на малюнку 1.2.

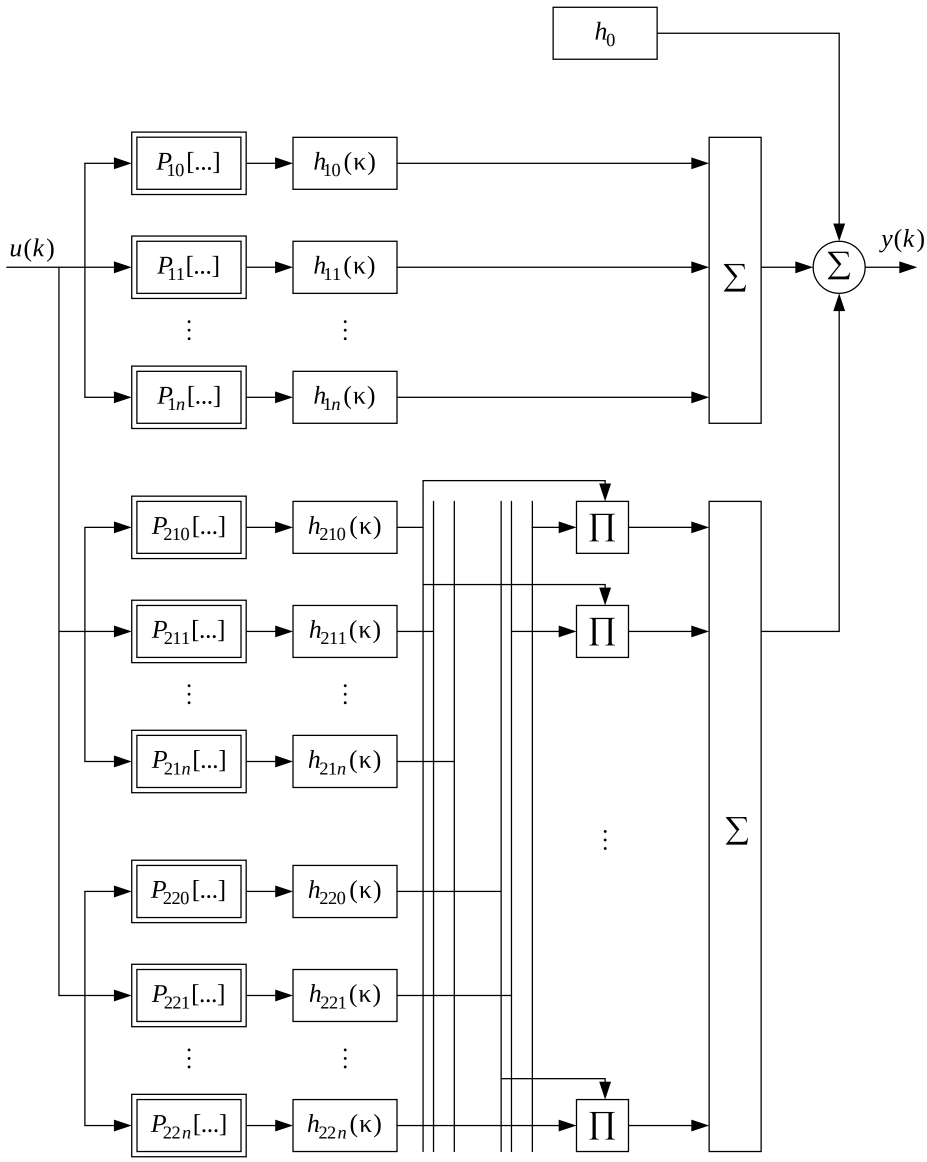

Рисунок 1.2 – Дискретна модель Урисона другого порядку

Дослідження і застосування даної моделі в задачах ідентифікації супроводжується значними труднощами. Незручність таких моделей найбільш чітко проявляється при ідентифікації нелінійного об’єкта в реальному часі. Труднощі застосування подібних моделей різко зростають, якщо досліджуваний нелінійний об'єкт є до того ж нестаціонарним.

Тому в ряді робіт [3, 4] досліджувалася нелінійна модель Вольтерра, що є окремим випадком моделі Урисона і заснована на поділі моделі на лінійну і нелінійну частини, які при наявності апріорної інформації можуть бути оцінені зі значно меншими обчислювальними затратами.

Отримана таким чином модель Вінера, що представляє собою послідовно з'єднані лінійну і нелінійну моделі, наведена на рис.1.3. Дана модель описується рівняннями:

$$x(k)=\frac{B(q^{-1})}{A(q^{-1})}u(k), \quad y(k)=P[x(k)], \tag{1.7}$$

де $B(q^{-1})=b_0+b_1q^{-1}+...+b_{n_b}q^{-n_b}$;

$A(q^{-1})=1+a_1q^{-1}+...+a_{n_a}q^{-n_a}$;

$P[x(k)]$ - деяка нелінійна функція.

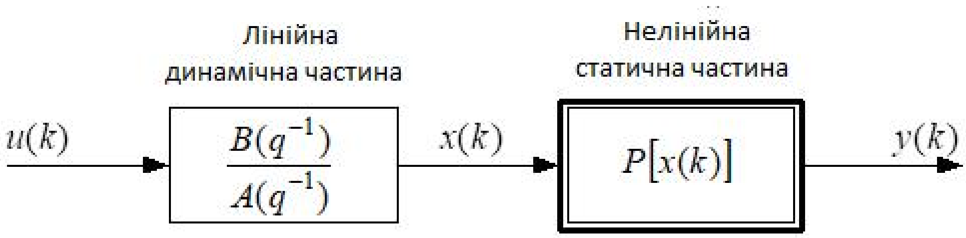

Рисунок 1.3 – Модель Вінера

Співвідношення (1.7) можуть бути записані наступним способом:

$$y(k)=P\left[\frac{B(q^{-1})}{A(q^{-1})}u(k)\right]. \tag{1.8}$$

Якщо в якості нелінійної функції P[·] обрана квадратична $P(x[k])=\gamma_0+\gamma_1 x(k)+\gamma_2 x^2(k)$, то рівняння (1.8) набирає вигляду:

$$y(k)=\gamma_0+\gamma_1\frac{B(q^{-1})}{A(q^{-1})}u(k)+\gamma_1\left[\frac{B(q^{-1})}{A(q^{-1})}u(k)\right]^2. \qquad (1.9)$$

Рівняння (1.9) є нелінійним відносно параметрів лінійної частини моделі. Однак в даний час методи оцінювання таких параметрів досить добре розроблені, тому рішення задачі ідентифікації може бути здійснено в реальному часі.

Модель Вінера допускає узагальнення, що полягає в використанні різних лінійних моделей (рис.1.4, рис.1.5), що застосовуються для опису лінійної і нелінійної частин, описувані відповідно рівняннями:

$$y(k)=\gamma_0+\frac{B_1(q^{-1})}{A_1(q^{-1})}u(k)+\left[\frac{B_2(q^{-1})}{A_2(q^{-1})}u(k)\right]^2; \qquad (1.10)$$

$$y(k)=\gamma_0+\frac{B_1(q^{-1})}{A_1(q^{-1})}u(k)+\left[\frac{B_2(q^{-1})}{A_2(q^{-1})}u(k)\right]\left[\frac{B_3(q^{-1})}{A_3(q^{-1})}u(k)\right]. \qquad (1.11)$$

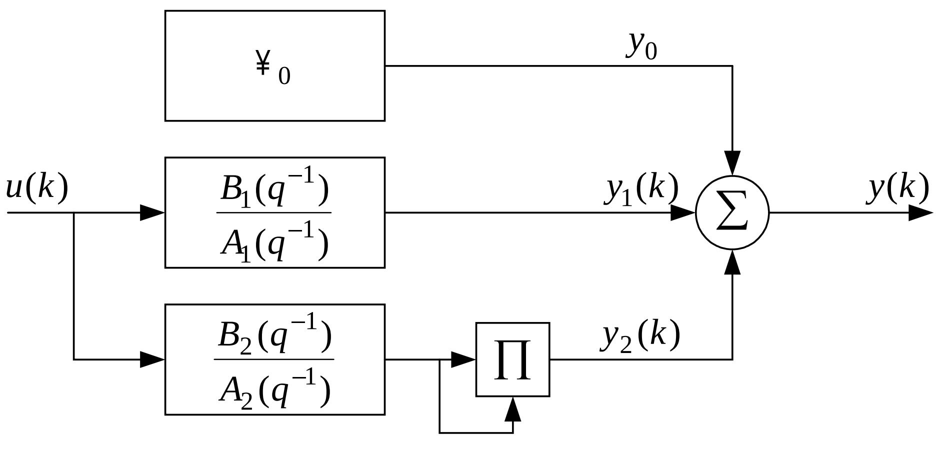

Рисунок 1.4 – Узагальнена модель Вінера, що описана рівнянням (1.10)

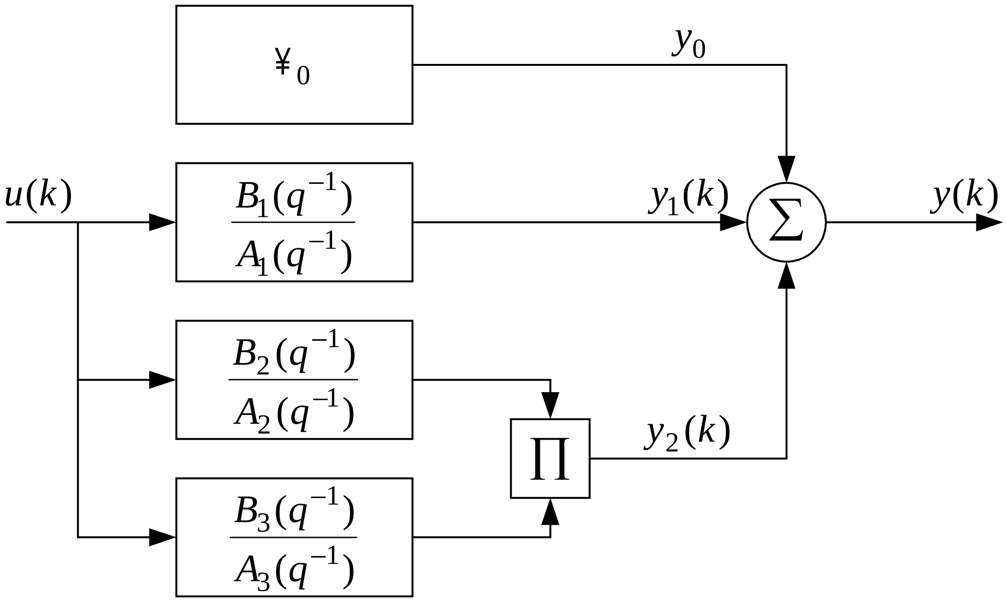

Рисунок 1.5 – Розширена модель Вінера, що описана рівнянням (1.11)

Модель Гаммерштейна, як і модель Вінера, містить послідовно з’єднані нелінійну і лінійну частини, однак, на відміну від останньої, вихідний сигнал моделі Гаммерштейна представляє минулий через лінійну частину сигнал, який є нелінійним перетворенням вхідного сигналу.

У момент часу k сигнал на виході нелінійної частини моделі $x(k)$ має вигляд:

$$x(k)=\gamma_0+\gamma_1 u(k)+\gamma_2 u^2(k)+...+\gamma_n u^n(k). \qquad (1.12)$$

Сигнал на виході моделі $y(k)$ отримується після пропускання $x(k)$ через лінійну частину, яка має вигляд:

$$\frac{B(q^{-1})}{A(q^{-1})}=\frac{b_0+b_1q^{-1}+...+b_mq^{-m}}{1+a_1q^{-1}+...+a_nq^{-n}} \quad m£n. \qquad (1.13)$$

Вперше модель Гаммерштейна для розв’язання задачі ідентифікації була запропонована Нарендра К. і Галлменом П. і мала вигляд, наведений на рисунку 1.6.

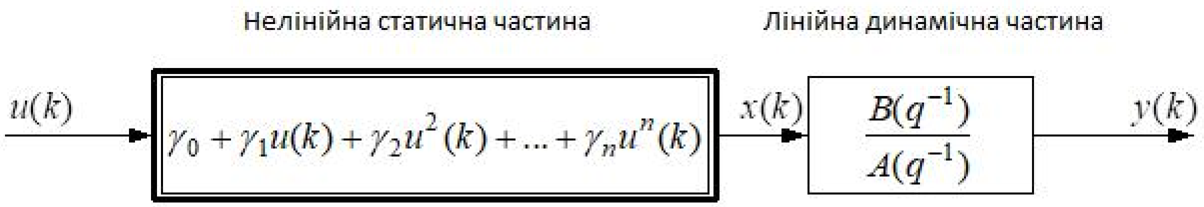

Рисунок 1.6 – Модель Гаммерштейна

У моделях Вінера і Гаммерштейна нелінійна частина відображає статику об’єкта і являє собою перетворення виду (1.1), а лінійна частина описує динаміку об’єкта.

Вінера і Гаммерштейна присвячено досить велику кількість робіт [1], в яких досліджуються як різні окремі випадки, так і можливі узагальнення даних моделей, що включають нестаціонарність, стохастичність, нелінійності, тощо. Серед таких узагальнень слід виділити модель Гаммерштейна-Вінера і узагальнену модель Гаммерштейна-Вінера [2], що представляють собою послідовне з’єднання моделі Гаммерштейна і моделі Вінера.

Квадратична модель Гаммерштейна-Вінера (рис.1.7) описується рівняннями:

$$\begin{cases} x_1(k)=\dfrac{B_1(q^{-1})}{A_1(q^{-1})}u(k),\;\; x_2(k)=\gamma_0+\gamma_1 x_1(k)+\gamma_2 x_1^2(k); \\ y(k)=\dfrac{B_2(q^{-1})}{A_2(q^{-1})}x_2(k). \end{cases} \tag{1.12}$$

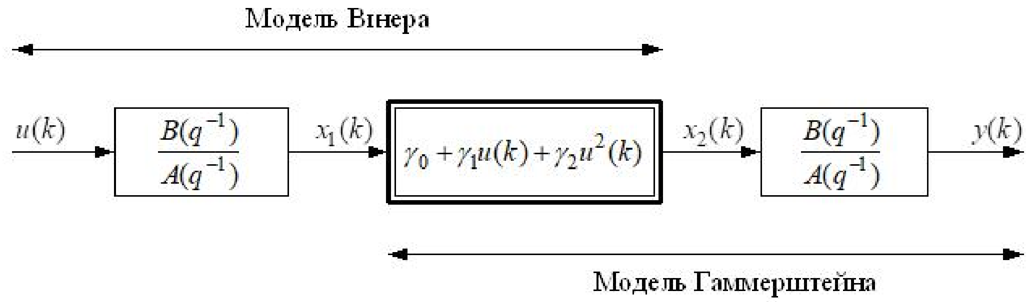

Рисунок 1.7 – Квадратична модель Гаммерштейна-Вінера

Виділивши постійну складову, лінійну і квадратичну частини, рівняння (1.12) можна привести до виду:

$$y_0=\gamma_0\frac{B_2(1)}{A_2(1)},\; y_1(k)=\gamma_1\frac{B_1(1)}{A_1(1)}\frac{B_2(1)}{A_2(1)}u(k),\; y_2(k)=\gamma_2\frac{B_2(q^{-1})}{A_2(q^{-1})}\left[\frac{B_1(q^{-1})}{A_1(q^{-1})}u(k)\right], \quad (1.13)$$
$$y(k)=y_0+y_1(k)+y_2(k).$$

Ввівши позначення $y_0=c_0$; $y_1(k)=B_1(1)/A_1(1)u(k)$, і враховуючи, що лінійна і квадратична частини моделі можуть мати різні передавальні функції, отримуємо узагальнену модель Гаммерштейна-Вінера (рис.1.8) описується рівняннями:

$$y_0=c_0,\; y_1(k)=\frac{B_1(1)}{A_1(1)}u(k),\quad y_2(k)=\frac{B_3(q^{-1})}{A_3(q^{-1})}\left[\frac{B_2(q^{-1})}{A_2(q^{-1})}u(k)\right]^2, \quad (1.14)$$
$$y(k)=y_0+y_1(k)+y_2(k).$$

Використання узагальненої моделі Гаммерштейна-Вінера дозволяє, в ряді випадків, істотно підвищити адекватність одержуваної математичної моделі реальному об’єкту [4]. Однак одержуваний при цьому результат не надто різниться від того результату, який дає застосування більш простої моделі, тобто або моделі Гаммерштейна, або моделі Вінера. Труднощі ж дослідження узагальненої моделі Гаммерштейна-Вінера різко зростають.

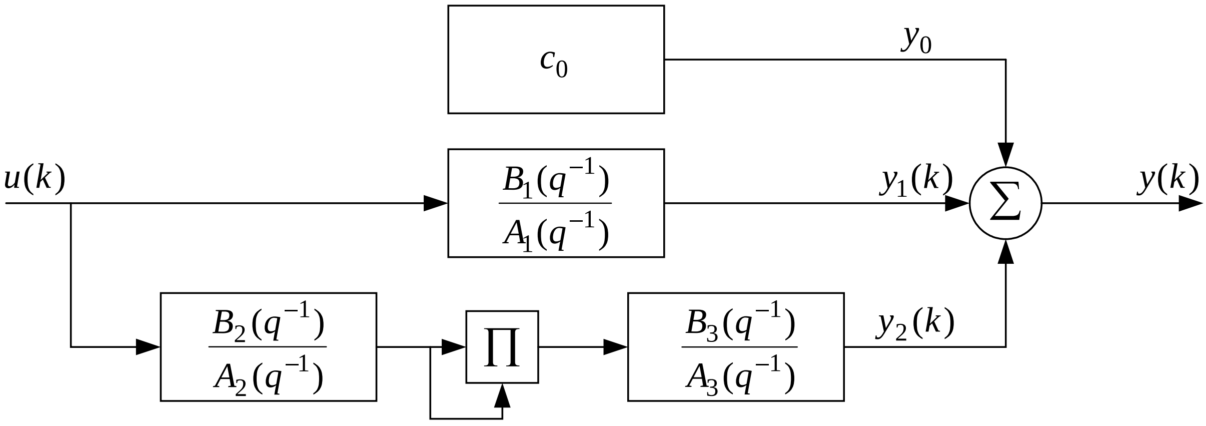

Рисунок 1.8 – Узагальнена модель Гаммерштейна-Вінера

1.1.4 Моделі ARMAX та NARMAX

Добре зарекомендували себе на практиці при апроксимації властивостей реальних лінійних фізичних систем статистичні моделі, такі як модель авторегресії (AR-модель), модель авторегресії з додатковими вхідними сигналами (ARX-модель), модель авторегресії змінного середнього з додатковими вхідними сигналами (ARMAX-модель), модель вихідний помилки (OE-модель), модель Бокса-Дженкінса (BJ-модель) та інші. Простота таких моделей і можливість використовувати для перевірки їх адекватності добре розроблені методи аналізу частотних характеристик є основними перевагами при їх використанні. Однак застосування лінійних моделей до складних нелінійних систем не забезпечує необхідної точності для цілей ідентифікації та управління.

Ґрунтуючись на поданні лінійних систем ARMAX моделлю, Леонарітіс І.Дж. і Біллінгз С.Д. [5] запропонували для опису нелінійних систем використовувати так звану нелінійну модель авторегресії змінного середнього з додатковими вхідними сигналами (NARMAX модель), що отримала подальший розвиток в роботах [6]:

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

де $y(k) \in R^{M\times 1}$ – вихідний сигнал об’єкта;

$u(k) \in R^{N\times 1}$ – вхідний сигнал об’єкта;

$e(k) \in R^{M\times 1}$ – помилка вимірювань;

$f[\cdot]$ – нелінійна функція перетворення $f\colon R^{(k_yM+k_uM+k_eM)\times 1} \to R^{M\times 1}$.

k_y, k_u, k_e - порядки запізнювання по вихідній, вхідного сигналів об’єкта і помилку вимірювань відповідно.

Як і в разі лінійних систем, нелінійні системи можуть бути представлені нелінійної NARX-моделлю:

$$y(k)=f[y(k-1),...,y(k-k_y),u(k-1),...,u(k-k_u)]+e(k) \quad (1.16)$$

Моделі (1.13) - (1.16) є непараметричних, проте використання теореми Стоуна-Вейерштрасса дозволяє їх параметризувати [7]. У цьому випадку модель (1.14) може бути представлена так:

$$\begin{aligned} y_i(k)&=c_0^{(i)}+\sum_{i_1=1}^{P} c_{i_1}^{(i)} z_{i_1}(k)+\sum_{i_1=1}^{P}\sum_{i_2=i_1}^{P} c_{i_1 i_2}^{(i)} z_{i_1}(k) z_{i_2}(k)+ \\ &+\sum_{i_1=1}^{P}\sum_{i_2=i_1}^{P}\cdots\sum_{i_l=i_{l-1}}^{P} c_{i_1 i_2 \dots i_l}^{(i)} z_{i_1}(k) z_{i_2}(k) \dots z_{i_l}(k)+e_i(k), \end{aligned} \quad (1.17)$$

де $P=Mk_y+Nk_u$;

$z_1(k)=y_1(k-1); z_2(k)=y_1(k-2);...; z_{Mk_y}(k)=y_M(k-k_y);$

$z_{Mk_y+1}(k)=u_1(k-1);\ z_{Mk_y+2}(k)=u_1(k-2);...; z_P(k)=u_N(k-k_u);$

c_j^i – коефіцієнти (параметри моделі).

Поліноміальний уявлення дозволяє перейти від завдання визначення нелінійної функції, що зв’язує вхідні і вихідні сигнали моделі, до задачі оцінювання параметрів, що входять лінійно в функцію. Якщо для опису нелінійного об’єкта з одним входом і одним виходом використовується NARX модель, що описується нелінійним рівнянням:

$$y(k)=f[y(k-1),y(k-2),u(k-1)]. \qquad (1.18)$$

Використання поліноміального уявлення (1.16) дозволяє отримати лінійне щодо шуканих параметрів рівняння:

$$\begin{gathered} y_i(k)=c_0+c_1y(k-1)+c_2y(k-2)+c_3y(k-3)+c_4y^2(k-1)+c_5y(k-1)y(k-2)+ \\ +c_6y(k-1)u(k-1)+c_7y^2(k-2)+c_8y(k-2)u(k-1)+c_9u^2(k-1). \end{gathered} \qquad (1.19)$$

Таким чином, завдання ідентифікації нелінійного об’єкта значно спрощується. Використання моделі також значно спрощує обчислення, так як призводить до лінійного щодо невідомих параметрів рівняння, яке може бути вирішено добре розробленими і досить повно дослідженими методами. Слід зазначити, що уявлення об’єктів у вигляді NARMAX або NARX моделей відіграють фундаментальну роль при дослідженні нелінійних об’єктів за допомогою ШНС.

У найбільш загальному вигляді, прийнятому в класичній теорії управління, опис стохастичних нелінійних систем в просторі станів розглядалося в роботі Нарендри К.С. і Партасараті К. [8]:

$$\begin{cases} z(k+1)=\Psi[z(k),u(k),f(k)]; \\ y(k)=\Phi[z(k),u(k),\varphi(k)], \end{cases} \qquad (1.20)$$

де $z(k)=[z_1(k),z_2(k),...,z_{k_z}(k)]^T$ – вектор станів системи;

$u(k)=[u_1(k),u_2(k),...,u_{k_u}(k)]^T$ – вектор вхідних сигналів;

$y(k)=[y_1(k),y_2(k),...,y_{k_y}(k)]^T$ – вектор вихідних сигналів;

$\Psi[\cdot], \Phi[\cdot]$ – деякі статичні нелінійні функції;

$f(k), \varphi(k)$ – шум процесу і шум вимірювань відповідно.

1.2 Нейромережевий підхід до ідентифікації

Ефективність створюємих систем управління реальними об’єктами в значній мірі залежить від якості використовуваних при цьому математичних моделей, які, з одного боку, повинні найбільш повно відображати властивості досліджуваного об’єкта, а з іншого – бути зручними для реалізації алгоритмів керування. Відсутність повної інформації про умови функціонування, властивості самих об’єктів і перешкод обумовлюють необхідність застосування при управлінні ними адаптивного підходу, що допускає можливість використання спрощених, зокрема, лінійних моделей. Хоча такий підхід і дозволяє в ряді випадків істотно зменшити апріорну невизначеність і реалізувати досить ефективну ідентифікацію, обмеження лінійними моделями не завжди забезпечує отримання необхідного результату. Тому більш ефективною є розробка систем управління на основі адаптивного підходу в поєднанні з методами теорії ШНМ.

Більшість реальних систем характеризуються нелінійними залежностями, складними для моделювання динамічними властивостями, наявністю неконтрольованих шумів і перешкод, що перешкоджають реалізації традиційних стратегій управління, оскільки, як сучасна (зокрема теорія адаптивного і оптимального управління), так і класична теорія управління в значній мірі базуються на ідеї лінеаризації систем [9].

Для практичного застосування даного підходу необхідна, перш за все, розробка математичних моделей. Однак математичне моделювання, що реалізовується на основі припущення про лінійність системи, може не відображати її дійсних фізичних властивостей. Навіть якщо вдається побудувати складні математичні моделі, точно відображають фізичні співвідношення між входом і виходом системи, вони можуть виявитися марними для цілей управління. Практично прийнятними можуть бути тільки моделі з низькою чутливістю по параметрам, що для нелінійних систем вкрай складно забезпечити [10-12].

Для практичного застосування алгоритмів управління необхідно, щоб вони були адаптивними, стійкими, нелінійними, а також простими для реалізації та розуміння. Саме з цих причин в даний час широке застосування в задачах управління отримали ШНМ, що володіють зазначеними вище властивостями.

Будучи альтернативою традиційним методам ідентифікації та управління, нейромережевий підхід заснований на повністю визначених ШНМ для вироблення необхідних сигналів управління. Високий інтерес до даного напрямку в теорії управління визначається наступними факторами:

- ШНМ – адаптивні системи, здатні навчатися вирішення складних завдань. Необхідною умовою якісного навчання є правильний вибір структури НММ і достатній обсяг інформації, який використовується при навчанні;

- нейромережеві технології управління дозволяють подолати багато труднощів, що виникають при роботі з нелінійними об’єктами або з об’єктами невідомої структури і нерозв’язні за допомогою звичайних методів адаптивного управління. Здатність ШНМ реалізовувати складні нелінійні відображення обумовлена застосуванням сигмоїдальних функцій активації (або деяких нелінійних функцій загального вигляду);

- високу швидкодію і надійність обумовлена високим ступенем паралельності ШНМ. Простота реалізації нейронних мереж і їх здатність до навчання роблять їх особливо привабливими при управлінні складними нелінійними об’єктами в реальному часі.

2 ПРИНЦИПИ ФОРМУВАННЯ НЕЙРОМЕРЕЖЕВИХ МОДЕЛЕЙ

2.1 Принципи формування нейромережевих моделей

Можна виділити два типи НММ об’єктів: моделі, отримані методами прямої і інверсної ідентифікації [13]. При побудові моделей, заснованих на прямих методах ідентифікації, використовують два різних підходи: «пророкування» (прогнозування) поведінки об’єкта і «імітація» або моделювання поведінки об’єкта. При моделюванні поведінки об’єкта доступною є лише інформація про значеннях вхідних сигналів $u(k), u(k\text{-}1),...$ і вихідних сигналів моделі $\hat{y}(k), \hat{y}(k\text{-}1),...$, при цьому сигнали на виході об’єкта не вимірюються. Отримана таким чином модель називається паралельної і описується рівнянням:

$$\hat{y}(k+1)=f\left(\hat{y}(k), \hat{y}(k\text{-}1),..., u(k), u(k\text{-}1),...\right). \quad (2.1)$$

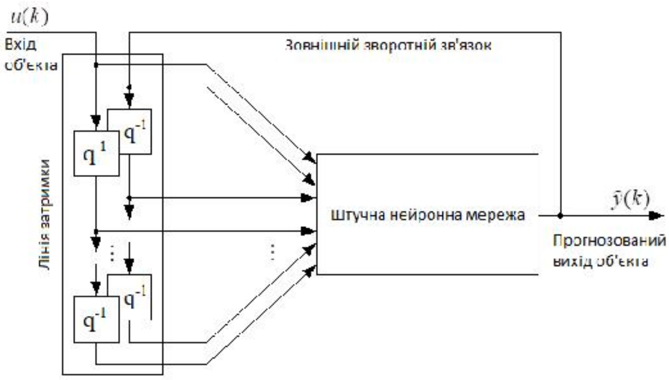

Рисунок 2.1 – Паралельна НММ

Паралельна модель (рисунок 2.1) може використовуватися для імітації поведінки об’єкта без підключення до реальної системи або коли датчики входять до складу самої моделі. Для діагностики стану системи необхідно порівняти значення сигналів на виході об’єкта і моделі і оцінити величину помилки.

При прогнозуванні поведінки об’єкта використовується послідовно-паралельна модель (рисунок 2.2). Значення сигналу на виході моделі $\hat{y}(k+1)$ розраховується на підставі значень вхідних $u(k), u(k\text{-}1), \ldots$ і вихідних $y(k), y(k\text{-}1), \ldots$ сигналів об’єкта згідно з рівнянням:

$$\hat{y}(k+1)=f\left(y(k), y(k\text{-}1), \ldots, u(k), u(k\text{-}1), \ldots\right). \quad (2.2)$$

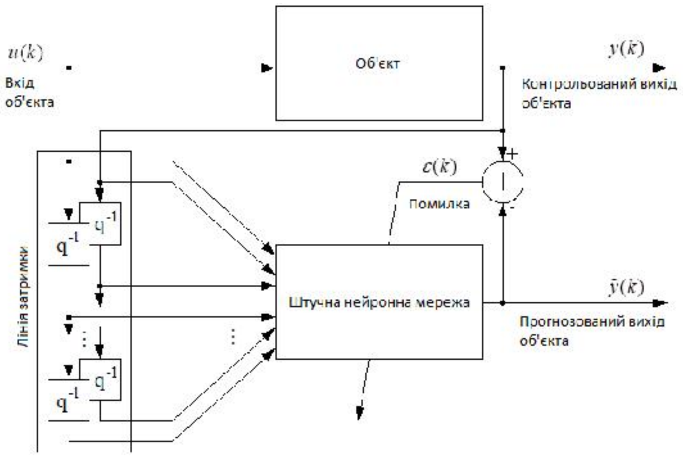

Рисунок 1.2 – Послідовно-паралельна НММ

Остання дозволяє передбачати поведінку об’єкта на будь-яку кількість тактів вперед, завдяки чому вона набула найбільшого поширення при вирішенні задач ідентифікації.

Якщо об’єкт описується рівнянням виду:

$$y(k+1)=f\left(y(k), y(k\text{-}1), \ldots, u(k)\right), \quad (2.3)$$

то для побудови системи управління можна використовувати інверсну модель (рисунок 1.3). Однак дана модель може бути застосована лише до обмеженого класу об’єктів, для яких запізнення по сигналу управління дорівнює одному такту.

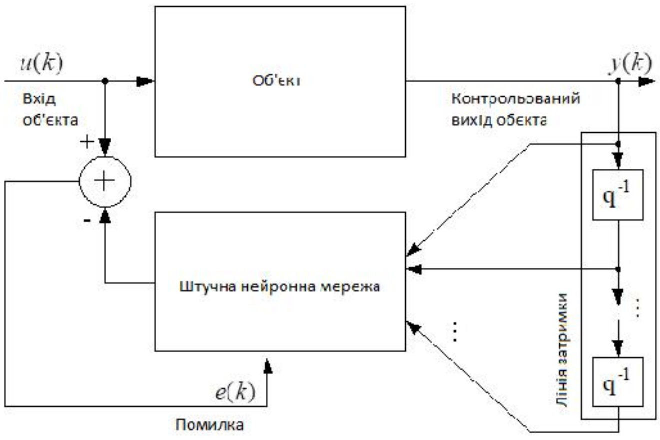

Рисунок 1.3 – Інверсна НММ

Найбільшого поширення для вирішення завдань ідентифікації та управління отримали ШНМ типу БП і РБМ [14-17]. Та обставина, що обидві ці мережі дозволяють апроксимувати з будь-якої заданої точністю будь-яку безперервну функцію, забезпечило їх широке застосування в задачах ідентифікації нелінійних об’єктів, а поєднання апроксимуючих властивостей зі здатністю швидко навчатися дозволяє використовувати їх для керування нелінійними динамічними об’єктами в реальному часі.

Обидві мережі використовують апроксимацію нелінійного оператора об’єкта (2.4) деякою системою базисних функцій $\{\Phi_i(p)\}$, реалізованої нейронами, що утворюють шари мережі, а завдання ідентифікації зводиться до навчання мережі, тобто налаштування параметрів мережі на основі пред’явлення навчальної вибірки, до складу якої входять вимірювані значення вхідних і відповідних вихідних змінних.

$$\hat{y}(k)=\sum_{i=1}^{N} w_i \Phi_i (p(k)), \qquad (2.4)$$

де w_i - вагові параметри мережі;

$p(k)=(y(k-1),...,y(k-k_y),u(k),...,u(k-k_u))^T$.

При побудові БП вибір виду активаційної функції відіграє істотну роль. Так як для навчання мережі зазвичай застосовуються різні методи оптимізації, які оперують як значеннями самої функції активації, так і значеннями її похідних, то необхідно, щоб використовувана функція задовольняла ряду умов: була обмеженою, монотонною і безперервно-диференційованою. Функції, що володіють перерахованими властивостями, називаються сигмоїдальними. До них відносяться:

- логістична:

$$f_{log}(u)=\frac{1}{1+e^{-\alpha u}};\ f'_{log}(u)=\alpha f_{log}(u)(1-f_{log}(u));\qquad(2.5)$$

- гіперболічний тангенс:

$$f_{th}(u)=\frac{e^{\alpha u}-e^{-\alpha u}}{e^{\alpha u}+e^{-\alpha u}};\ f'_{th}\left(u\right)=1-f_{th}^{2}(u).\qquad(2.6)$$

Особливо важливу роль такі функції відіграють при моделюванні нелінійних залежностей між вхідними та вихідними змінними.

Як уже зазначалося, БП та РБМ є універсальними апроксиматорами і дозволяють апроксимувати з будь-якою заданою точністю будь-яку безперервну функцію, за умови, що існує достатня кількість прихованих шарів в мережі. Це твердження справедливо і для мереж тільки з одним прихованим шаром. Необхідною умовою є те, що функції активації нейронів прихованого шару $f(\cdot)$ повинні бути безперервними, обмеженими і непостійними. Ці вимоги досить м'які, причому сигмоїдальні функції (2.5) і (2.6) є лише одним з можливих варіантів активаційних функцій, що задовольняють цим умовам. Даний теоретичний результат є логічним обґрунтуванням використання РБМ для моделювання нелінійних систем, тому що гарантує, що мережі з одним прихованим шаром завжди буде достатньо для подання будь-якої довільної неперервної функції. Однак відкритим залишається питання про вибір кількості нейронів в прихованому шарі, яке забезпечить задану точність рішення. Крім того, викладені результати засновані на припущенні, що значення ваг в мережі задані коректно.

В даний час існує велика кількість методів налаштування параметрів ШНМ, що відрізняються як динамічними властивостями, так і обчислювальною складністю. Навчання БП складається в налаштуванні ваг, що характеризують силу взаємодії між нейронами, і зазвичай виконується на скінечній навчальній множині, що містить *P* навчальних пар. Кожна навчальна пара включає вектор вхідних сигналів $p(k)(1 \le k \le P)$ і відповідний йому цільової (бажаний) вихідний вектор $d(k)$.

РБМ мають високу швидкість навчання і завдяки механізму адаптації структури мережі при їх навчанні не виникає проблем із зупинкою в локальних мінімумах. Однак треба зазначити, що число нейронів в шаблонному шарі РБМ експоненціально залежить від розмірності вхідного простору, що при використанні паралельної або паралельно-послідовної моделей об’єктів може привести до досить громіздкою структурою мережі і збільшити час отримання результату. Зміна структури РБМ зазвичай здійснюється її поступовим ускладненням шляхом додавання нових нейронів, кожний раз, коли при появі чергового вхідного сигналу виникає помилка ідентифікації, що перевищує допустиму. Параметрична ідентифікація (навчання мережі) складається в налаштуванні її вагових коефіцієнтів і зводиться до мінімізації зазвичай квадратичного функціоналу помилки ідентифікації [18].

Крім розглянутих вище мереж, в даний час при вирішенні задач ідентифікації та управління використовуються мережі типу CMAC, засновані на моделі, яка описує процеси управління рухом, що відбуваються в мозочку, запропонованої Дж. Албусом. Особлива привабливість мереж CMAC обумовлена високою швидкістю їх навчання (це досягається застосуванням спеціальних способів кодування інформації), що дозволяє використовувати дані мережі при апаратній реалізації керуючих систем на базі мікроконтролерів [19]. При цьому зазвичай обмежуються побудовою систем управління об’єктами з розмірністю не вище третього порядку, тому що збільшення розмірності досліджуваного об’єкта призводить як до ускладнення кодування інформації (появі багатовимірних матриць асоціацій), так і до різкого зростання кількості необхідних обчислень [20].

2.2 Аналіз алгоритмів навчання ШНМ

Навчання нейромережі – ітеративний процес. На кожній ітерації обчислюються мережеві виходи для одного (або більше) зразка в навчальному наборі, і коригуються мережеві ваги з метою зменшення помилки між фактичним мережевим виходом $(y^{L}_{i,p})(i=1,...N^{L})$ і цільовим виходом $(d_{i,p})$ для даного зразка. При використанні підходящої мережної архітектури і алгоритму навчання мережеві ваги прагнуть до значень, при яких мережевий вихід стає прийнятно близьким до цільового виходу для кожного зразка в наборі навчання [21].

2.2.1 Вибір критерію якості навчання

Точність результату на кожній ітерації оцінюється з використанням функції помилки E (або енергетичної функції $J(w)$):

$$E=\sum_{p=1}^{P} E_p \,, \qquad (2.7)$$

де E_p – частковий внесок зразка p в повну мережеву помилку E .

Вибір функції помилки має настільки ж істотний вплив на ефективність БП, як і вибір алгоритму навчання. Найбільш широко використовують функцією помилки є квадратична функція помилки [22]:

$$E=\frac{1}{2}\sum_{p=1}^{P}\sum_{i=1}^{N^L}\left(d_{i,p}-y_{i,p}^L\right)^2 . \qquad (2.8)$$

Нормалізована її версія, яка називається середньоквадратичної функцією помилки:

$$E=\frac{1}{2PN^L}\sum_{p=1}^{P}\sum_{i=1}^{N^L}\left(d_{i,p}-y_{i,p}^L\right)^2 . \qquad (2.9)$$

Перевага останньої полягає в тому, що вона нечутлива як до числа зразків в наборі навчання, так і до числа нейронів у вихідному шарі мережі, що дозволяє використовувати її для порівняння в різних завданнях навчання.

Особливістю функцій (2.8) і (2.9) є те, що зменшення E може бути пов’язано зі збільшенням числа помилково класифікуються зразків; фактично, зменшення E до мінімально прийнятного рівня будь-який з цих функцій помилки не обов’язково відповідає мінімальному числу нерозпізнаних образів. Це пов’язано особливостями поверхні функціоналу помилки БП – локальними мінімумами і багатовимірними "плато" [23].

БП застряг в локальному мінімумі, коли величина мережевий помилки E для послідовних періодів навчання стабілізується на порівняно високому рівні. Локальні мінімуми, як відомо, утворюються в специфічних тестових завданнях. Однак вони відсутні, якщо навчається завдання є лінійно нероздільні або число прихованих вершин мережі збігається з кількістю зразків в наборі навчання, або число зразків менше або дорівнює числу параметрів моделі. У кожному з цих випадків відбувається навчання мережі з обраної архітектурою. Однак, на жаль, жоден з наявних результатів не містить практичних рекомендацій для реальних додатків.

Використовують альтернативні функції помилки, що дозволяють поліпшити характеристики збіжності алгоритмів навчання БП. Один з таких підходів полягає в застосуванні функції помилки, що дозволяє мережі вийти з локального мінімуму, зокрема, функції помилки з взаємної ентропією:

$$E=-\sum_{p=1}^{P}\sum_{i=1}^{N^L}\ln\left[\left(y_{i,p}^L\right)^{d_{i,p}}\left(1-y_{i,p}^L\right)^{1-d_{i,p}}\right]. \qquad (2.10)$$

При використанні (2.10), градієнти помилки для погано помітних зразків значно вище, ніж при використанні квадратичної і середньоквадратичної функцій помилки, що дозволяє мережі швидше рухатися в плоских областях простору ваг (тобто при перетині плато).

Друга стратегія полягає у введенні обмежень в функцію помилки, таким чином, щоб мережа не сходилася до поганих зразків в просторі ваг. Одним з варіантів такої функції помилки є показова функція помилки:

$$E=\frac{1}{2}\sum_{p=1}^{P}\sum_{i=1}^{N^L}e^{-\alpha\left(y_{i,p}^L-d_{i,p}+\beta\right)\left(d_{i,p}+\beta-y_{i,p}^L\right)}, \qquad (2.11)$$

де α и β – позитивні, визначені користувачем параметри.

Значення параметру β задає допустимий рівень помилки для кожного елемента $(d_{i,p})$ в наборі навчання. Моллєр рекомендує задавати початкове значення $\beta=0.9$, і потім ділити його навпіл щоразу, коли на кожному виході мережі $(y_{i,p}^L)$ помилка досягає допустимого рівня [24]. Параметр α задає крутизну експоненти, збільшуючи помилку для тих мережевих виходів, які мають неприпустимий рівень помилок, який визначається параметром β. При $\alpha\to\infty$ не відбувається зменшення мережевий помилки E, що відповідає збільшенню числа неправильно класифікованих зразків. Установка досить великого значення коефіцієнта α може привести до значного зменшення швидкості збіжності алгоритму, тому що діапазон допустимих рішень обмежений. Хороші результати дає $\alpha=1.0$, хоча оптимальне значення α залежить від конкретного завдання.

Навчання БП є успішним при досягненні мінімального значення помилки, що визначається з умов конкретного завдання. Слід зазначити, що досягнення мінімального значення помилки часто є небажаним, тому що призводить до явища перенавчання БП і погіршення його фільтруючих і узагальнюючих властивостей. Виняток становлять завдання апроксимації функцій, для яких потрібна висока точність одержуваних рішень.

Перший теоретично доведений алгоритм для настройки ваг при навчанні БП відомий як алгоритм зворотного поширення помилки (ЗП). Сьогодні ЗР може розглядатися як еталонний тест, з яким порівнюються всі інші методи навчання.

2.2.2 Алгоритм зворотного поширення помилки

Термін «зворотне поширення» відноситься до процесу, за допомогою якого можуть бути обчислені похідні функціоналу помилки за параметрами мережі. Цей процес може використовуватися в поєднанні з різними стратегіями оптимізації. При використанні квадратичної функції помилки вираз для градієнта критерію якості функціонування для нейронів вихідного шару має вигляд [25]:

$$\begin{aligned}\frac{\partial E}{\partial w_{ik}^{L}} &= \sum_{p=1}^{P}\sum_{k=1}^{N^L}\frac{\partial E}{\partial y_{k,p}^{L}}\frac{\partial y_{k,p}^{L}}{\partial a_{k,p}^{L}}\frac{\partial a_{k,p}^{L}}{\partial w_{ik}^{L}} = \frac{\partial}{\partial w_{ik}^{L}}\left(\frac{1}{2}\sum_{p=1}^{P}\sum_{k=1}^{N^L}\left(d_{k,p}-y_{k,p}^{L}\right)^2\right) = \\ &= -\sum_{p=1}^{P}\sum_{k=1}^{N^L}\left(d_{k,p}-y_{k,p}^{L}\right)\frac{\partial y_{k,p}^{L}}{\partial w_{ik}^{L}}.\end{aligned} \tag{2.12}$$

Правило функціонування вихідного шару L:

$$y_{k,p}^{L} = f^{L}\left(\sum_{l=0}^{N^{L-1}} w_{kl}^{L}\, y_{l,p}^{L-1}\right), \tag{2.13}$$

де $k = \overline{1, N^{L}}$

Приймаючи до уваги, що:

$$\frac{\partial y_{k,p}^{L}}{\partial w_{ij}^{L}} = \begin{cases} 0, & k \neq i \\ f'^{L}(a_{i,p}^{L})\, y_{j,p}^{L-1}, & k = i \end{cases} \tag{2.14}$$
$$i = \overline{1, N^{L}}, \qquad j = \overline{1, N^{L-1}},$$

де $i = \overline{1, N^{L}}$,

$j = \overline{1, N^{L-1}}$.

Отримаємо:

$$\frac{\partial E}{\partial w_{ij}^{L}} = -\sum_{p=1}^{P}\left(d_{i,p} - y_{i,p}^{L}\right) f'^{L}(a_{i,p}^{L})\, y_{j,p}^{L-1}. \tag{2.15}$$

Ввівши позначення для локальної помилки вихідного шару:

$$\delta_{i,p}^{L} = \left(d_{i,p} - y_{i,p}^{L}\right) f'^{L}(a_{i,p}^{L}), \;\; i = \overline{1, N^{L}} \tag{2.16}$$

Отримаємо:

$$\frac{\partial E}{\partial w_{ij}^{L}} = -\sum_{p=1}^{P} \delta_{i,p}^{L}\, y_{j,p}^{L-1}, \tag{2.17}$$

де $i = \overline{1, N^{L},}\,;$

$j = \overline{0, N^{L-1}}$.

Перейшовши до шару $L-1$, маємо:

$$\frac{\partial E}{\partial w_{ij}^{L-1}} = \sum_{p=1}^{P}\sum_{k=1}^{N^L}\frac{\partial E}{\partial y_{k,p}^{L}}\frac{\partial y_{k,p}^{L}}{\partial a_{k,p}^{L}}\frac{\partial a_{k,p}^{L}}{\partial f_{j,p}^{L-1}}\frac{\partial f_{j,p}^{L-1}}{\partial a_{j,p}^{L-1}}\frac{\partial a_{j,p}^{L-1}}{\partial w_{ij}^{L-1}} =$$
$$= -\sum_{p=1}^{P}\sum_{k=1}^{N^L}\left(d_{k,p} - y_{k,p}^{L}\right) f'^{L}(a_{k,p}^{L})\, w_{jk}^{L}\, f'^{L-1}(a_{j,p}^{L-1})\, y_{j,p}^{L-2} = -\sum_{p=1}^{P}\delta_{i,p}^{L-1}\, y_{j,p}^{L-2}, \quad (2.18)$$

де $\delta_{i,p}^{L-1}$ – локальна помилка шару $L-1$.

$$\delta_{i,p}^{L-1} = \sum_{k=1}^{N^L}\left(d_{k,p} - y_{k,p}^{L}\right) f'^{L}(a_{k,p}^{L})\, w_{jk}^{L}\, f'^{L-1}(a_{j,p}^{L-1}) = f'^{L-1}(a_{j,p}^{L-1})\sum_{k=1}^{N^L}\delta_{k,p}^{L} w_{jk}^{L};$$
$$i = \overline{1, N^{L-1}}.$$

Для шарів $L-2,\ L-3, ..., 1$ обчислення часткових похідних критерію за елементами матриць вагових коефіцієнтів виконується аналогічно. У підсумку, узагальнена формула (узагальнене дельта правило) має вигляд:

$$\frac{\partial E}{\partial w_{ij}^{r}} = -\sum_{p=1}^{P}\delta_{i,p}^{(r-1)}\, y_{j,p}^{(r-1)}, \qquad r = \overline{1, L},\ \ i = \overline{1, N^r},\ \ j = \overline{0, N^{r-1}}, \quad (2.19)$$

де r - номер шару, а $\delta_{i,p}^{r}$ – локальна помилка шару r.

$$\delta_{i,p}^{r} = f'^{r}(a_{i,p}^{r})\sum_{k=1}^{N^{r+1}}\delta_{k,p}^{r+1} w_{ki}^{r+1}, \qquad r = \overline{1, L-1};$$
$$\delta_{i,p}^{L} = \left(d_{i,p} - y_{i,p}^{L}\right) f'^{L}(a_{i,p}^{L}), \qquad i = \overline{1, N^L}.$$

У стандартній реалізації алгоритму ЗП для мінімізації цільової функції (функції помилки) використовується градієнтний метод пошуку (заснований на локальної лінійної апроксимації цільової функції), згідно з яким на кожній ітерації *k* обчислюється градієнт функціоналу $\nabla f(w(k))$ і коригуються мережеві ваги за правилом:

$$\begin{cases} w(k+1) = w(k) + \Delta w(k); \\ \Delta w(k) = -\eta \nabla f(k),\ \eta > 0, \end{cases} \tag{2.20}$$

де η - норма навчання, зазвичай встановлюється в діапазоні (0,1).

$$\nabla f(k) = \left(\frac{\partial E\left(w(k)\right)}{\partial w_1(k)} \quad \frac{\partial E\left(w(k)\right)}{\partial w_2(k)} \quad \ldots \quad \frac{\partial E\left(w(k)\right)}{\partial w_N(k)} \right)^T .$$

Істотним недоліком даного алгоритму є його повільна збіжність. Крім того, обмеження, що накладаються даним методом на властивості цільової функції (по-перше, цільова функція повинна бути досить гладкою, по-друге, цільова функція не повинна збільшуватися швидше квадратичної) і вибір величини норми навчання η, різко скорочують коло завдань, для яких можна застосувати алгоритм ЗП. У зв'язку з цим, було запропоновано численні його модифікації, які полягають як у застосуванні змінюється в часі параметра η:

$$\eta(k+1) = \begin{cases} \rho\eta(k), \text{якщо}\, E\left(w(k+1)\right) < E\left(w(k)\right); \\ \sigma\eta(k), \text{якщо}\, E\left(w(k+1)\right) \geq E\left(w(k)\right), \end{cases} \tag{2.21}$$

де $\rho > 1,\ 0 < \sigma < 1$.

2.2.3 Метод Ньютона

У градієнтному методі основою є ідея локальної лінійної апроксимації цільової функції. Якщо ж цільова функція $f(w(k))$ двічі диференційована, то ефективніше використовувати її квадратичну апроксимацію в точці $w(0)$.

Ітераційна процедура пошуку мінімуму (метод Ньютона) має вигляд:

$$w(k+1) = w(k) - G^{-1}(k)\nabla f(k) \qquad (2.22)$$

де
$$G(k) = \begin{bmatrix} \frac{\partial^2 E\left(w(k)\right)}{\partial w_1^2(k)} & \frac{\partial^2 E\left(w(k)\right)}{\partial w_1(k) w_2(k)} & \cdots & \frac{\partial^2 E\left(w(k)\right)}{\partial w_1(k) w_N(k)} \\ \frac{\partial^2 E\left(w(k)\right)}{\partial w_2(k) w_1(k)} & \frac{\partial^2 E\left(w(k)\right)}{\partial w_2^2(k)} & \cdots & \frac{\partial^2 E\left(w(k)\right)}{\partial w_2(k) w_N(k)} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 E\left(w(k)\right)}{\partial w_N(k) w_1(k)} & \frac{\partial^2 E\left(w(k)\right)}{\partial w_N(k) w_2(k)} & \cdots & \frac{\partial^2 E\left(w(k)\right)}{\partial w_N^2(k)} \end{bmatrix}.$$

Для квадратичної функції метод Ньютона сходиться за один крок. Чим ближче цільова функція до квадратичної, тим швидше сходиться метод Ньютона.

Для навчання нейронних мереж застосування методу Ньютона, заснованого на вираженні (2.12), недоцільно через [26]:

- метод можна використовувати тільки, якщо матриця Гессе позитивно визначена;
- метод вимагає істотних обчислювальних витрат і значного обсягу пам’яті для зберігання даних;
- при методу використанні необхідні аналітичні вирази для перших і других похідних в кожній точці $w(k)$.

Використання різних стратегій для регулювання позитивної визначеності матриці Гессе і для вибору прийнятною довжини кроку на кожній ітерації дозволяє модифікувати метод Ньютона таким чином, щоб надати йому властивість глобальної збіжності і знизити його обчислювальні витрати. Так для гладких сильно опуклих функцій демпфірований метод Ньютона (2.13), заснований на регулюванні довжини кроку, глобально сходиться [27]:

$$w(k+1) = w(k) - \lambda(k)G^{-1}(k)\nabla f(k) = w(k) - l(k), \quad (2.23)$$

де $\lambda(k)$ – довжина кроку.

Ефективною є інша модифікація методу Ньютона (названа методом Левенберга-Марквардта), в якій напрям пошуку відрізняється від задається методом Ньютона. В даному методі до апроксимуючої функції додається квадратичний штраф за відхилення від точки $w(k)$, а нова точка $w(k+1)$ визначається з умови мінімуму функції 28]:

$$f_k(w) + \alpha(k)\|w - w(k)\|^2 / 2, \quad (2.24)$$

де $f_k(w) = f(w(k)) + \left(w - w(k)\right)^T \nabla f\left(w(k)\right) + 0.5\left(w - w(k)\right)^T G(k)\left(w - w(k)\right)$

Що призводить до методу:

$$w(k+1) = w(k) - \left[G(k) + \alpha(k)I\right]^{-1} \nabla f(k). \quad (2.25)$$

При $\alpha(k) = 0$ метод переходить в метод Ньютона, при $\alpha(k) \to \infty$ напрямок пошуку прагне до антиградієнта. Таким чином, метод Левенберга-Марквардта (2.25) являє собою компроміс між цими двома методами. За рахунок вибору $\alpha(k)$ можна домогтися глобальної збіжності методу. Крім того, даний метод придатний не тільки для опуклих функцій, тоді як в методі (2.23) потрібно позитивна визначеність матриці Гессе.

Однак в розглянутих модифікаціях методу Ньютона кожна ітерація (як і в основному методі Ньютона) вимагає великих обчислювальних витрат (обчислення других похідних, рішення систем лінійних рівнянь), а швидкість збіжності далеко від мінімуму не висока.

2.2.4 Квазіньютоновські методи

В основі цих методів лежить ідея відновлення квадратичної апроксимації цільової функції за значеннями її градієнтів у ряді точок. Тим самим методи об’єднують гідності градієнтного методу (не потрібно обчислення матриці других похідних) і методу Ньютона (швидка збіжність внаслідок використання квадратичної апроксимації).

Ці методи мають загальну структуру:

$$w(k+1) = w(k) - \alpha(k)H(k)\nabla f(k), \tag{2.26}$$

де матриця перераховується рекурентним способом на основі інформації, отриманої на $k-$й ітерації, так що $H(k) - G^{-1}(k) \to 0$.

Основна ідея квазіньютоновських методів полягає в побудові апроксимації безпосередньо для оберненої матриці $G^{-1}(k)$.

Найбільш поширені такі формули для перерахунку матриці $H(k+1)$ [29]:

а) метод Давидона-Флетчера-Пауэлла (ДФП):

$$H(k+1) = H(k) - \frac{H(k)\,p(k)p^T(k)H(k)}{p^T(k)H(k)p(k)} + \frac{s(k)s^T(k)}{p^T(k)s(k)}, \quad H(0) > 0;$$

б) метод Бройдена:

$$H(k+1) = H(k) - \frac{\big(s(k) - H(k)\,p(k)\big)\big(s(k) - H(k)\,p(k)\big)^T}{p^T(k)\big(s(k) - H(k)\,p(k)\big)}, \qquad H(0) > 0;$$

в) метод Бройдена-Флетчера-Шенно (БФШ):

$$H(k+1) = H(k) + \left(1 + \frac{p^T(k)H(k)p(k)}{s^T(k)p(k)}\right)\frac{s(k)s^T(k)}{s^T(k)p(k)} -$$
$$- \frac{s(k)p^T(k)H(k) + H(k)p(k)s^T(k)}{s^T(k)p(k)}, \qquad H(0) > 0,$$

де $p(k) = \nabla f(k+1) - \nabla f(k)$;

$s(k) = w(k+1) - w(k)$.

Головний недолік квазіньютоновських методів в порівнянні, наприклад, з методом сполучених градієнтів полягає в необхідності зберігати і перераховувати матрицю $H(k)$ розмірності $n \times n$, що для великих n вимагає великого обсягу пам'яті і виконання великого числа арифметичних операцій.

2.2.5 Методи сполучених градієнтів

У градієнтному методі на кожному кроці не використовується інформація, отримана на попередніх ітераціях. Методи, в яких нове наближення залежить від s попередніх називаються s-кроковими:

$$w(k+1) = \phi(w(k), ..., w(k-s+1)) \qquad (2.27)$$

Градієнтний метод и метод Ньютона – однокрокові методи. Методи сполучених градієнтів відносяться до багатокрокових методів і, на відміну від методів Ньютона, не вимагають зберігання матриці Гессе [30]. Ці методи засновані на тому, що послідовність взаємно пов'язаних напрямків пошуку, яка задовольнить умові $t(k) = -H^{-1}(k)\nabla f(k)$, може бути знайдена ітераційно, без використання матриці Гессе, відповідно до правила:

$$t(k+1) = -\nabla f(k+1) + \beta(k)t(k), \qquad (2.28)$$

де початковий напрямок пошуку $t(0) = -\nabla f(0)$.

Тоді процес пошуку мінімуму цільової функції набирає вигляду:

$$w(k+1) = w(k) + \alpha(k)t(k), \tag{2.29}$$

де $\alpha(k) = \alpha$;

α – значення, яке мінімізує функцію $E(w(k) + \alpha\, t(k))$.

Відмінності методів сполучених градієнтів полягають в способі обчислення $\beta(k)$. Найбільш відомими з них є:

- метод Флетчера- Рівса:

$$\beta(k) = \nabla f\,(k+1)^{T} \nabla f\,(k+1) \big/ \nabla f\,(k)^{T} \nabla f\,(k) \tag{2.30}$$

- метод Полака-Ріб'ера:

$$\beta(k) = \left(\nabla f\,(k+1) - \nabla f\,(k)\right)^{T} \nabla f\,(k+1) \Big/ \nabla f\,(k)^{T} \nabla f\,(k) \tag{2.31}$$

Для квадратичних функцій методу сполучених градієнтів дає точку мінімуму за число ітерацій, що не перевершує n. Для неквадратичних функцій властивість кінцівки втрачається [31]. Тому для них вводиться процедура поновлення – час від часу крок робиться не по формулі (2.28), а як в початковій точці, тобто по антиградієнту. Ефективно проводити оновлення через число ітерацій, що дорівнює розмірності простору n. При цьому метод сполучених градієнтів з оновленням має властивість глобальної збіжності [32].

3 ПОБУДОВА МОДЕЛІ

Ідентифікація динамічних об’єктів в загальному випадку полягає у визначенні їх структури і параметрів за спостережуваним даними – вхідного впливу і вихідним значенням.

Застосування нейронних мереж для ідентифікації потребує розв’язання певних завдань ще на підготовчих етапах [12]:

- вибір конфігурації мережі: від кількості шарів та нейронів в них залежить гнучкість системи, а відповідно здатність до апроксимації складних функцій з заданою точністю, проте значне збільшення кількості нейронів призводить до перенавчання мережі та виникнення «прокляття розмірності»;
- вибір алгоритму навчання: при невдалому виборі методу та алгоритму навчання час необхідний на апроксимацію з заданою точністю може значно зрости, крім того може виникнути параліч мережі або навчання зупиниться через потрапляння до локального мінімуму;
- формування навчальної вибірки: навчальна вибірка повинна містити достатню кількість даних про всі екстремуми функції, при цьому вхідні дані повинні бути нормалізовані та рівномірно розподілені в просторі. Недотримання цієї вимоги призводить до некоректного навчання мережі, коли значення похибки буде різне для різних ділянок характеристики.

Задача ідентифікації полягає оцінюванні функції $f(\cdot)$ за вимірюваннями вхідних $u(k)$ та вихідних $y(k)$ змінних. Слідуючи з цього загальна схема ідентифікації динамічного об’єкта з використанням штучної нейронної мережі має вигляд, що зображений на рисунку 3.1. На вхід ШНМ подаються як вхідні параметри, так і вихідні. Також у схемі врахована похибка, що може бути похибкою вимірювань, тощо.

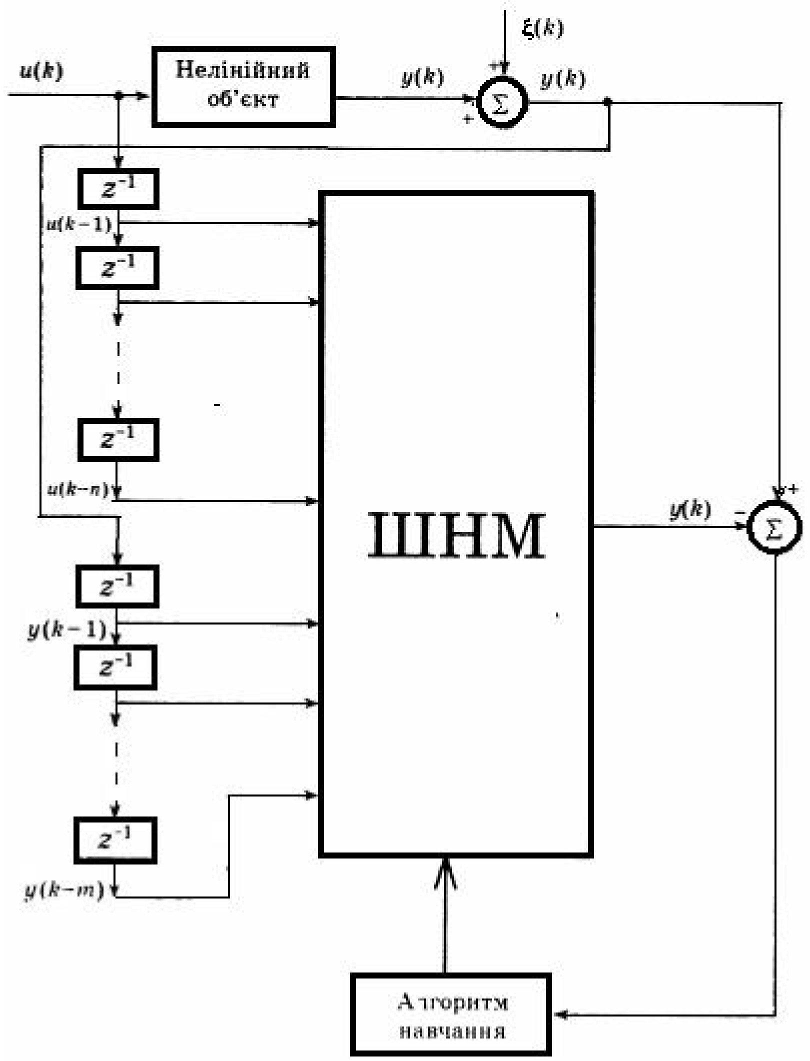

Рисунок 3.1 – Схема ідентифікації динамічного об’єкту

3.1 Вибір мережі та її конфігурації

Як зазначалося у другому розділі найбільшого поширення для вирішення завдань ідентифікації отримали штучні нейронні мережі типу багатошаровий персептрон та радіально-базисні мережі. Ці мережі є універсальними апроксиматорами.

Після проведення аналізу предметної області було прийнято рішення використовувати РБМ для ідентифікації нелінійних динамічних об’єктів. РБМ мають високу швидкість навчання і завдяки механізму адаптації структури мережі при їх навчанні не виникає проблем із зупинкою в локальних мінімумах.

У загальному випадку під РБМ розуміється двошаровий мережу без зворотних зав'язків, яка містить прихований шар радіально симетричних нейронів. Математичною основою РБМ є метод потенціальних функцій, розроблений, який позволяє представити деяку функцію f(x) у вигляді суперпозиції потенціальних чи базисних функцій $f_i(x)$:

$$f(x)=\sum_{i=1}^{N} a_i f_i(x)=a^T f(x), \qquad (3.1)$$

де $a_i(t) = (a_1, a_2, \dots , a_N)^T$ – вектор параметрів, які слід знайти;

$f_i(x) = (f_1(x), f_2(x), \dots , f_N(x))^T$ – вектор базисних функцій.

В якості базисних функцій вибираються деякі функції відстані між векторами:

$$f_i(x)=f(\|x-c_i\|) \qquad (3.2)$$

Вектори c_i називають центрами базисних функцій. Функції $f_i(x)$ вибираються невід'ємними та зростаючими при збільшенні $\|x-c_i\|$. У якості міри зближення векторів x та c_i обирається евклідова метрика чи манхетенська.

РБМ складаються з більшої кількості нейронів, ніж стандартні мережі з прямою передачею сигналів і навчанням методом зворотного поширення помилки, але на їх створення потрібно значно менше часу. Ці мережі найбільш ефективні, коли є велика кількість навчальних векторів [12-17].

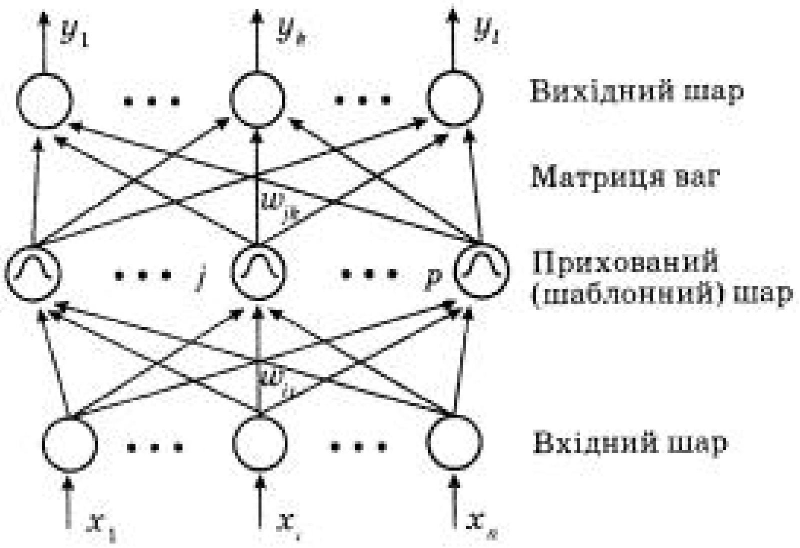

Рисунок 3.2 – Структура радіально-базисної мережі

У РБМ інформація передається з вхідного шару до прихованого, який називається шаблонним та має p нейронів. Кожний нейрон шаблонного шару отримує повну інформацію про вхідні сигнали x та обчислює функцію:

$$f_i = f((x-c_i)^T R^{-1} (x-c_i)), \qquad (3.3)$$

де $i=\overline{1,p}$;

x – вектор сигналів $N\times1$;

c_i – вектор центрів $N\times1$;

R – вагова матриця.

Особливістю даної мережі є наявність радіально-базисного шаблонного шару, в якому аналізується відстань між вхідним вектором і центром, представленим у вигляді вектору у вхідному просторі. Вектор центрів визначається за навчальною вибіркою і зберігається в просторі ваг від вхідного шару до шаблонного шару.

Нейрон шаблонного шару представлений на рисунку 3.3. Обробку інформації, що до нього надходить, можна розділити на два етапи:

- Обчислення відстані між представленим чином x і вектором центрів c_i з урахуванням метрики і норми матриці R;
- Відстань перетворюється нелінійною активаційною функцією.

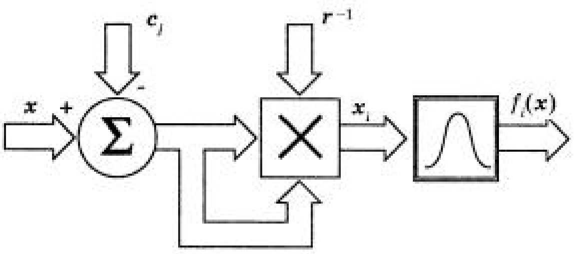

Рисунок 3.3 – Нейрон шаблонного шару

У якості функції активації обирають наступні:

- гаусова функція (рисунок 3.4):

$$f(x)=\exp\left\{-\frac{(x-c)^2}{2\sigma^2}\right\}. \tag{3.2}$$

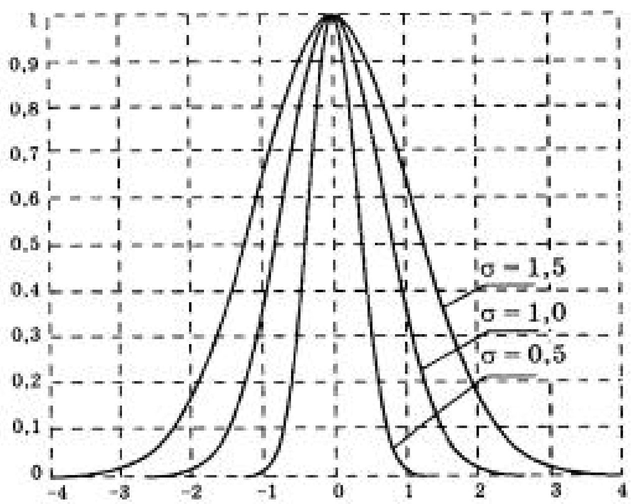

Рисунок 3.4 – Гаусова функція

- мультиквадратична функція (рисунок 3.5):

$$f(x)=\left[\frac{(x\text{-}c)^2}{\sigma^2}+a^2\right]^{\frac{1}{2}}. \qquad (3.3)$$

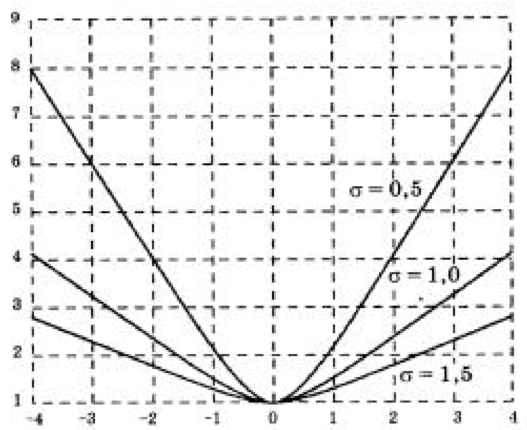

Рисунок 3.5 – Мультиквадратична функція

- зворотна мультиквадратична функція (рисунок 3.6):

$$f(x)=\left[\frac{(x\text{-}c)^2}{\sigma^2}+a^2\right]^{-\frac{1}{2}}. \qquad (3.3)$$

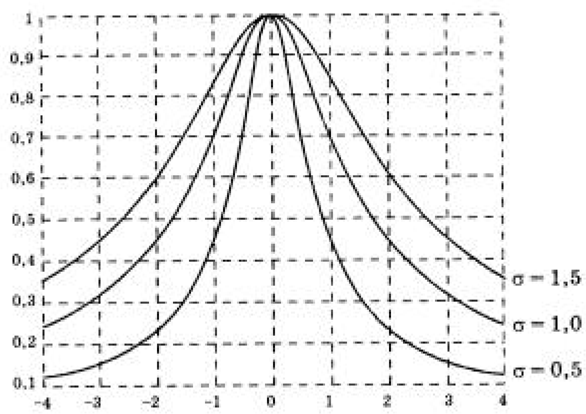

Рисунок 3.6 – Зворотна мультиквадратична функція

- сплайн функція (рисунок 3.7):

$$f(x)=x^2\log(x)\,. \qquad (3.4)$$

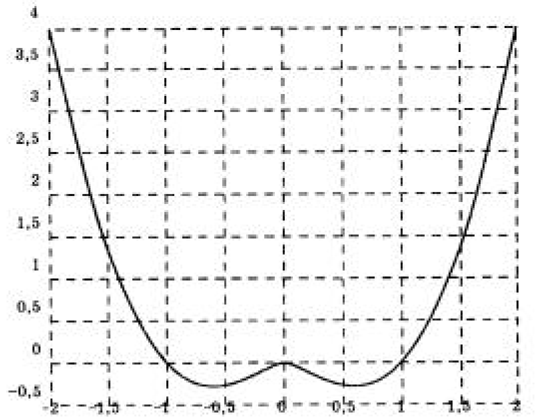

Рисунок 3.7 – Сплайн функція

- функція Коши (рисунок 3.8):

$$f(x)=(1+|x|)^{-1}\,. \qquad (3.5)$$

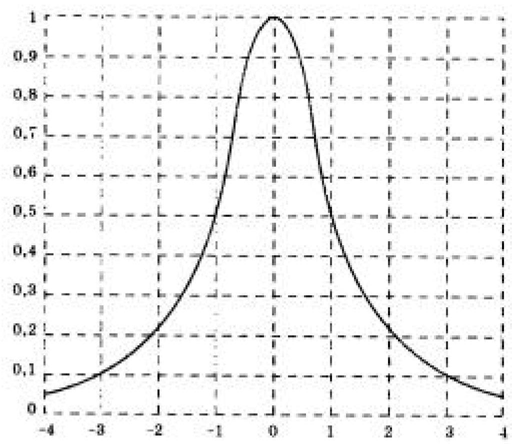

Рисунок 3.8 – Функція Коши

Усі ці функції добре підходять для моделювання нейронів прихованого шару мережі. Найчастіше використовуємою є гаусова функція [18]. Тож при моделюванні мережі буде використана саме вона.

Розглянемо параметр R^{-1}, що входить до рівняння 3.3. Це норма матриці, що встановлює положення осей у просторі. В загальному випадку матриця може бути представлена у вигляді:

$$R^{-1}=\begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1p} \\ r_{21} & r_{22} & \ldots & r_{2p} \\ \ldots & \ldots & \ldots & \ldots \\ r_{p1} & r_{p2} & \ldots & r_{p1} \end{bmatrix}. \tag{3.6}$$

Вагову матрицю R^{-1} також називають зворотною коваріаціонною матрицею. Елементи цієї матриці дорівнюють:

$$r_{ij}=\sigma_{ij}^{-2}, \tag{3.7}$$

де $i,j=\overline{1,p}$;

$\sigma_{ij}^{-2}=\sigma_{ji}^{-2}$;

σ_{ij}^{-2} – деякі керовані параметри.

Вагових параметри нейронів вихідного шару визначаються за допомогою вирішення лінійних алгебраїчних рівнянь:

$$y^{*}=F\omega, \tag{3.8}$$

де $y^{*}=\left(y_1^{*},y_2^{*},...,y_N^{*}\right)^T$ – заданих значень функції, що апроксимується, у опорних точках $c_1,c_2,...,c_N$;

$f_{ij}(x)=f(\|x-c_i\|)$ – базисні функції;

$\mu = \left(\mu_1, \mu_2, ..., \mu_N\right)^T$ – вектор вагових параметрів мережі.

Схема ідентифікації нелінійного динамічного об’єкта з використанням РБМ зображена на рисунку.

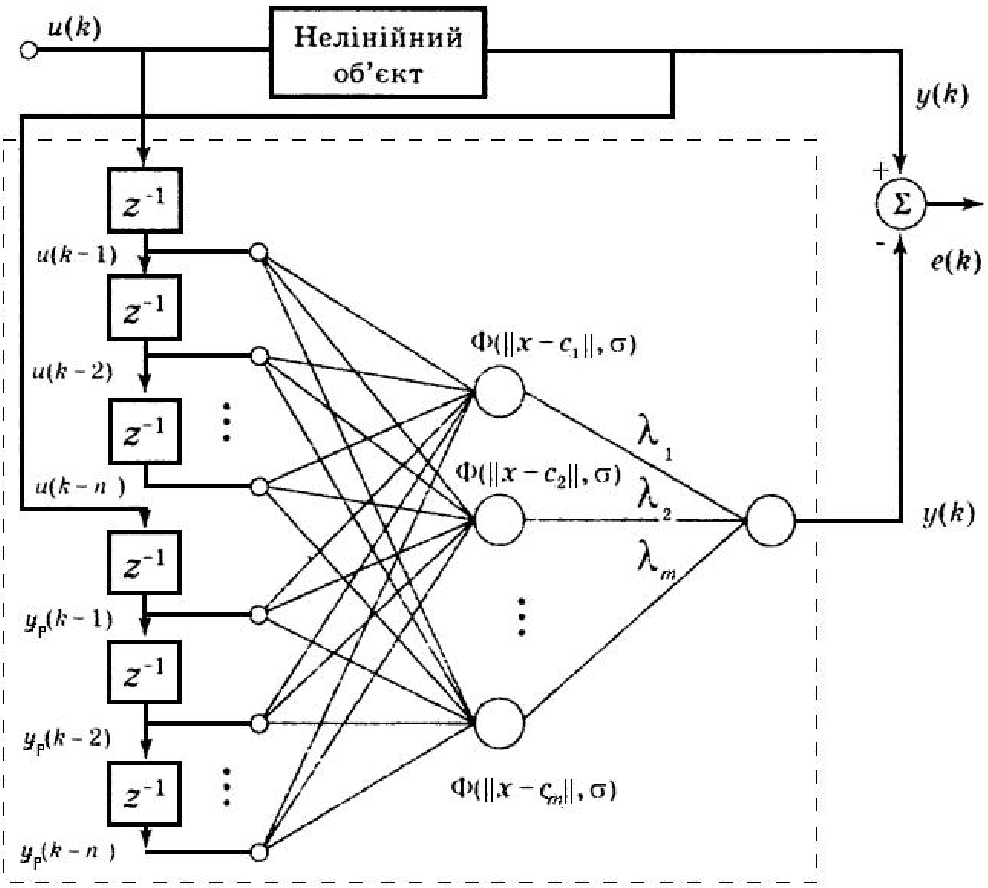

Рисунок 3.9 – Схема ідентифікації з використанням РБМ

3.2 Вибір алгоритму навчання

Згідно з першим пунктом розділу, РБМ характеризують три типи параметрів [12]:

- лінійні вагові параметри вихідного шару ω_{ij}, що входять в опис мережі лінійно;
- центри c_i – нелінійні параметри прихованого шару;
- відхилення (радіуси базисних функцій) σ_{ij} – нелінійні параметри

прихованого шару.

Навчання мережі полягає у визначенні цих параметрів. Є три наступні варіанти:

- задаються центри і відхилення, а визначаються тільки ваги вихідного шару;
- визначаються шляхом самонавчання центи та відхилення, а для корекції вагів вихідного шару використовується навчання з учителем;
- визначаються усі параметри за допомогою навчання з учителем.

Перші два варіанта використовуються у мережах, що використовують базисні функції з жорстко заданим радіусом(відхиленням). Третій варіант передбачає використання будь-яких базисних функцій.

3.2.1 Вибір параметрів центрів та відхилень

Центри визначають точки, через які повинна проходити функція, яка апроксимується. Оскільки велика навчальна вибірка призводить до сповільнення процесу навчання, в РБ-мережах широко використовується кластеризація образів, при якій схожі вектори об’єднуються в кластери, що подаються потім в процесі навчання одним вектором. в даний час існує досить велика кількість ефективних алгоритмів кластеризації.

Одним з даних алгоритмів є алгоритм к-середніх. Він направляє кожен образ в кластер, який має найближчі до даного центр. Якщо кількість центрів заздалегідь задано і визначено, алгоритм, обробляючи на кожному такті вхідний вектор мережі, формує в просторі входів мережі центри кластерів. Кандидатами в центри є всі виходи прихованого шару, проте в результаті роботи алгоритму буде сформована підмножина найбільш істотних виходів.

Як уже зазначалося параметр σ_i, що входить в формули для функцій перетворення c_i, визначає відхилення щодо центру. Варіюючи параметри c_i та σ_i, необхідно перекрити весь простір образів, не залишаючи пустот.

Використовуючи метод к-найближчих сусідів, визначаються к сусідів центру c_i та вираховується середнє значення $\hat{c}_i$. Значення $\hat{c}_i$ від c_i служить підставою для вибору параметру σ_i. На практиці часто виправдовує себе вибір:

$$\sigma = \frac{\max(c_i - c_k)}{\sqrt{2p}} , \qquad (3.9)$$

де p – кількість нейронів прихованого шару.

Вагові коефіцієнти вираховуються з співвідношення (3.8).

Якщо якість апроксимації є незадовільною, то вибір параметрів та розрахунок вагів повторюється, доки якість не стане задовільною.

3.2.2 Самонавчання параметрів центрів

Налаштування параметрів центрів відбувається наступним чином. Компонентам вагового вектору надаються випадкові початкові значення. Корекція компонент вектору центрів, розташованого найбільш близько до пред’явленого навчально образу відбувається за алгоритмом:

$$\sigma = c_i(k+1) = c_i(k) + \alpha(k)\partial_{ij}[x(k) - x_i(k)] , \qquad (3.10)$$

де $i = \overline{i,p}$;

$i = \arg\min_j \{d_j(k)\}$;

$d_j(k) = \left\| x(k) - c_j(k) \right\|$.

Коефіцієнт посилення $\alpha(k) \in (0,1)$, що змінюється у часі, задається співвідношенням:

$$\alpha(k)=\frac{\alpha(k\text{-}1)}{\sqrt{1+\text{int}\left(\frac{k}{N+p+1}\right)}}\ ,\qquad (3.11)$$

де int(x) – ціла частина x.

Вирахувані значення центрів застосовуються для знаходження функції перетворення f_i та обчислення виходів мережі.

3.2.3 Навчання мережі з учителем

Як і в разі будь-якого навчання з учителем мережі послідовно пред’являються навчальні пари $\left(x_i,y_i^*\right)$ $i=\overline{1,k}$, де x_i та y_i^* – вхідний та бажаний вихідний вектори і-й навчальної пари. На виході з’являється вектор $y_i=(y_{1i},y_{2i},...,y_{Li})^T$. Цей вектор порівнюється з вимагаємим y_i^* та вираховується відстань неузгодженості $\left(y_i\text{-}y_i^*\right)$ використовувана в заздалегідь обраному опуклому функціоналі якості, мінімізація якого дає необхідні значення елементів вагової матриці.

Як функціонал, що мінімізується, приймемо квадратичний виду:

$$I=\sum_{i=1}^{k}\left(y_i^*\text{-}y_i\right)^T\left(y_i^*\text{-}y_i\right)=\sum_{i=1}^{k}\left(y_i^*\text{-}Wf(x_i)\right)^T\left(y_i^*\text{-}Wf(x_i)\right).\qquad (3.12)$$

Всі пари $\left(x_i,y_i^*\right)$ $i=\overline{1,k}$ утворюють матриці вхідних сигналів X(k) розмірності $N\times k$ та матриці вимагаємих вихідних сигналів $Y^*(k)$ розмірності $L\times k$. РБМ описується наступним матричним рівнянням:

$$Y(k)=WF(k),\qquad (3.13)$$

де $Y(k)$ – ($L\times k$)-матриця вхідних сигналів;

W – ($L \times p$)-матриця вагів;

$Y^*(k)$ – ($L \times k$)-матриця вимагаємих сигналів.

Тоді критерій може буди записаний наступним чином:

$$I = tr\left[\left(Y(k) - WF(k)\right)^T \left(Y(k) - WF(k)\right)\right], \qquad (3.14)$$

де $tr[\cdot]$ – слід матриці.

Мінімізація відбувається шляхом вирішення системи лінійних алгебраїчних рівнянь:

$$\frac{\partial I}{\partial \omega_{ij}} = 0 . \qquad (3.15)$$

При пред'явленні нової навчальної пари оцінка матриці вагових коефіцієнтів має вигляд:

$$W(k+1) = Y(k+1)\left(F(k+1)\right)^T \left(F(k+1)\left(F(k+1)\right)^T\right)^{-1} . \qquad (3.16)$$

4 ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ

Імітаційне моделювання є необхідним етапом дослідження ефективності як обраної архітектури ІНС, так і вживаного алгоритму навчання мережі.

В даний час найбільш масовим напрямком нейрокомпьютирінга є моделювання нейронних мереж на персональних комп’ютерах. Успіхи в моделюванні дозволяють створювати навчальні програми, що дають користувачеві повне уявлення про ІНС і їх можливості.

Однак незалежно від використовуваного пакету результатом експериментального дослідження є визначення областей найбільш ефективного застосування того чи іншого типу ІНС і розробка рекомендацій щодо їх практичного використання.

4.1 Вибір програмного середовища для моделювання нейромереж

В даний час популярні і широко використовуються наступні програмні нейросимулятори:

- Stuttgart Neural Network Simulator, дозволяє емулювати роботу досить великої кількості алгоритмів нейронних мереж, є найбільш універсальним і багатофункціональним серед вільно розповсюджуваних програмних нейросимуляторов;
- Trajan, на жаль, доступна тільки демо-версія, яка дозволяє проектувати моделі нейронних мереж, проводити їх навчання і спостерігати за результатами роботи створеного проекту. У пакеті реалізовані алгоритм зворотного поширення, алгоритм сполучених градієнтів, Левенберга-Марквардта, швидкого поширення, Delta-Bar-Delta, що дозволяє використовувати версію в освітніх цілях;
- NeuroPro, розроблений в Інституті обчислювального моделювання СО РАН місто Красноярськ. У даній програмі реалізовані широкі можливості щодо спрощення нейронних мереж на основі обчислюваних показників

значущості, що дозволяє відсіяти малозначні, "шумові" вхідні сигнали, і залишити елементи, дійсно необхідні для правильного вирішення завдання, а потім привести нейронну мережу до логічно прозорого виду, коли по структурі мережі стає можливим побудувати явний алгоритм вирішення задачі;

- найбільш повні можливості з дослідження властивостей алгоритмів навчання ШНМ надає програмний пакет MATLAB. Входить до його складу Neural Network Toolbox являє собою пакет програм, орієнтований на вирішення широкого спектра завдань з використанням нейромережевих алгоритмів. У ньому передбачена реалізація 15 різновидів нейронних мереж, а також можливість створення користувацьких мереж практично будь-якої конфігурації [33].

У даній роботі дослідження алгоритмів навчання радіально базисної мережі проводилися саме в середовищі MATLAB.

4.2 Дослідження швидкості збіжності та надійності алгоритмів навчання РБМ

Велика кількість існуючих алгоритмів мінімізації функціоналу, використовуваних при навчанні мереж, ускладнює їх вибір. Конкретні практичні рекомендації можуть бути отримані шляхом імітаційного моделювання різних алгоритмів. Слід зазначити, що вибір конкретної стратегії навчання і структури мережі визначається розв’язуваної завданням. В даному розділі проведено порівняльний аналіз ефективності алгоритмів навчання РБМ при вирішенні тестових завдань апроксимації тригонометричних функцій sin(x), tan(x), sin(x)cos(2x) (рис.4.1) [16, 17]. Для навчання мереж використовувалися наступні алгоритми:

- методи сполученого градієнта (в якості опції одновимірного пошуку використовувалася функція Брента):

a) метод Флетчера-Рівса;

b) метод Полака-Ріб'ера.

- квазіньютоновський метод з модифікацією Бройдена-Флетчера-Гольдфарба-Шанно (в якості опції одновимірного пошуку використовувалася функція перебору з поверненнями);
- метод Левенберга-Марквардта.

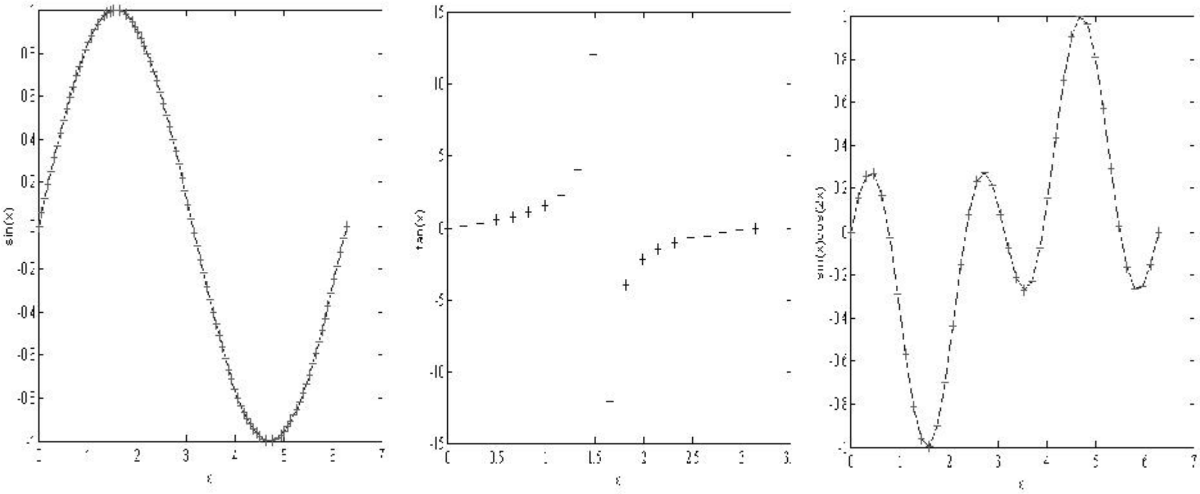

Рисунок 4.1 – Завдання апроксимації тригонометричних функцій

Всі алгоритми виконувалися в пакетному режимі по 100 разів з новими початковими значеннями ваг і зміщень. Максимальне число допустимих ітерацій – 10000. При вирішенні завдань апроксимації тригонометричних функцій, використовувалася гаусова функція активації для нейронів у шаблонному шарі, а для нейронів у вихідному шарі – лінійна. У якості оцінки використовувалися два аспекти роботи алгоритмів – «глобальна надійність» (тобто наскільки алгоритм схильний зупинятися в локальних мінімумах) і швидкість навчання [22].

Чисельної оцінкою глобальної надійності алгоритму є відсоток успішної збіжності (УЗ) до вирішення з заданим ступенем точності, а швидкості збіжності – середнє число ітерацій (СЧІ), необхідне для досягнення рішення при заданій точності.

Таблиця 4.1 – Оцінка алгоритмів навчання при моделюванні функції sin(x)

Метод	E=0.1		E=0.01		E=0.001		E=0.0001	
	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС
БФГШ	3.09	100	29.4	100	49.6	99	121	99
Флетчера-Рівса	4.54	100	98.5	89	202	89	1675	88
Полака-Ріб’ера	4.37	100	319	88	2501	91	7145	48
Левенберга-Марквардта	2.49	100	10.3	88	11	94	12	91

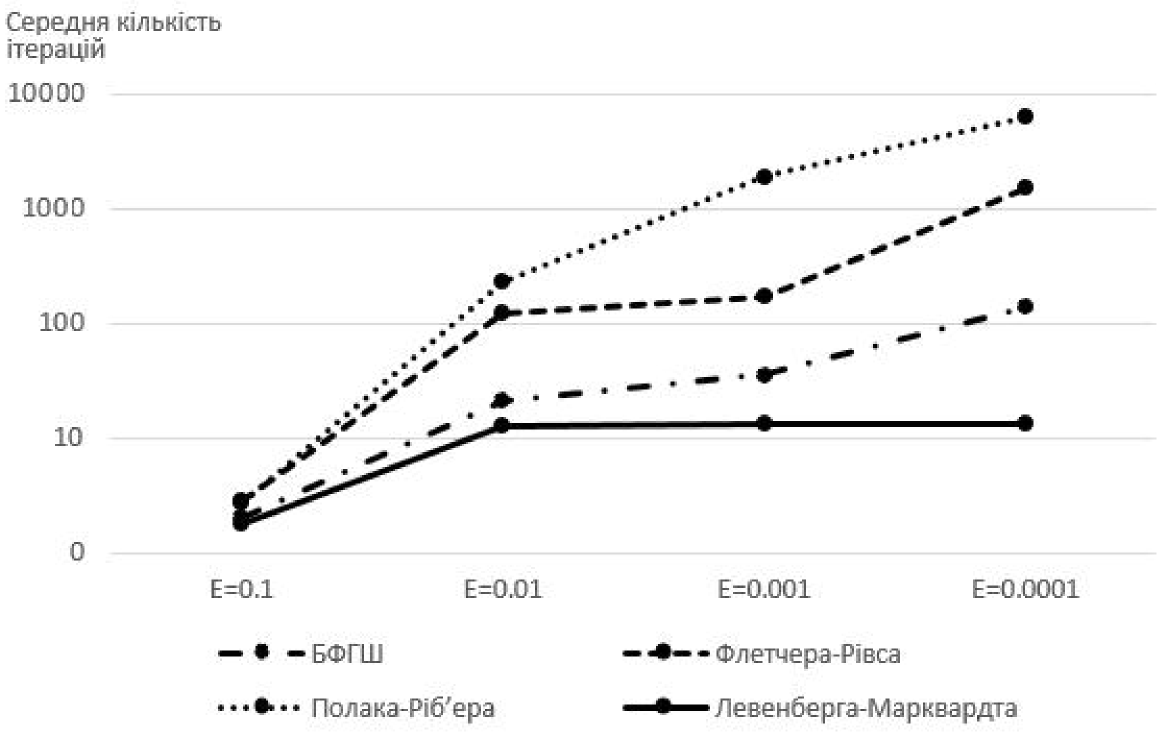

Рисунок 4.2 – Оцінка швидкості збіжності алгоритмів навчання при моделюванні функції sin(x)

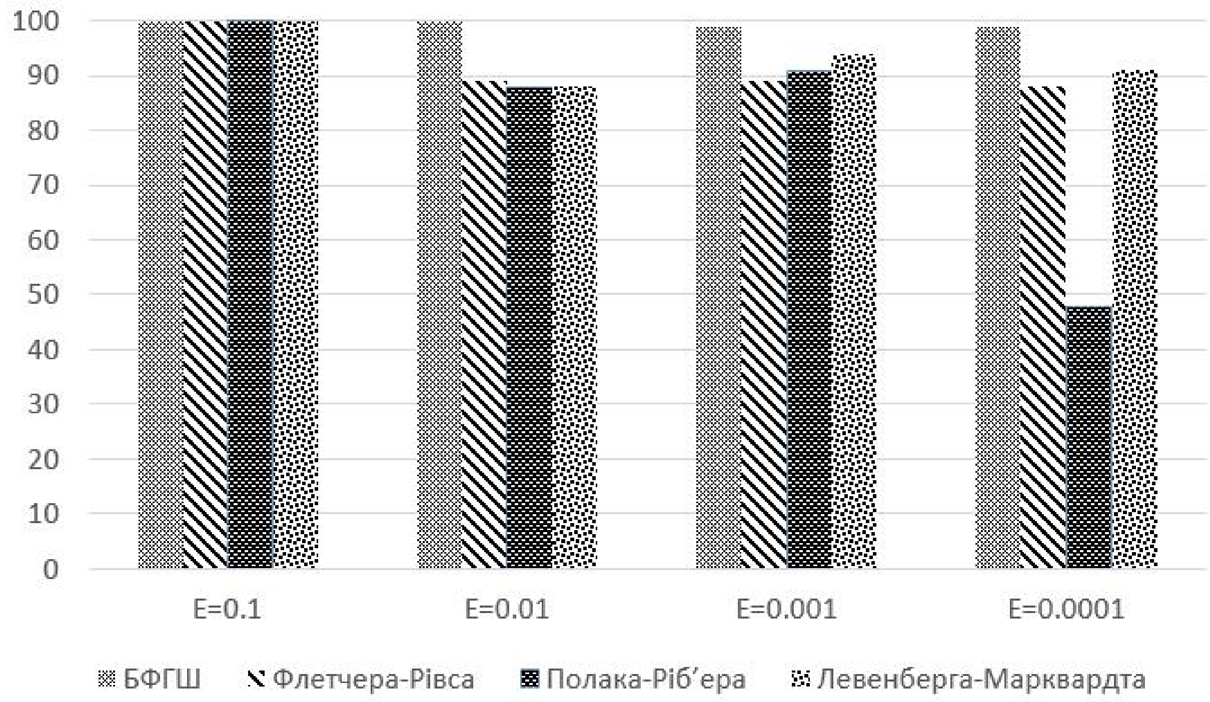

Рисунок 4.3 – Оцінка глобальної надійності алгоритмів навчання при моделюванні функції sin(x)

Таблиця 4.2 – Оцінка алгоритмів навчання при моделюванні функції tan(x)

Метод	E=0.1		E=0.01		E=0.001		E=0.0001	
	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС
БФГШ	67.1	100	141.23	87	577	61	1685	31
Флетчера-Рівса	591	100	1156	97	4598	19	-	-
Полака-Ріб’ера	591	100	1156	97	5786	24	-	-
Левенберга-Марквардта	68.45	100	106	99	156	96	457	75

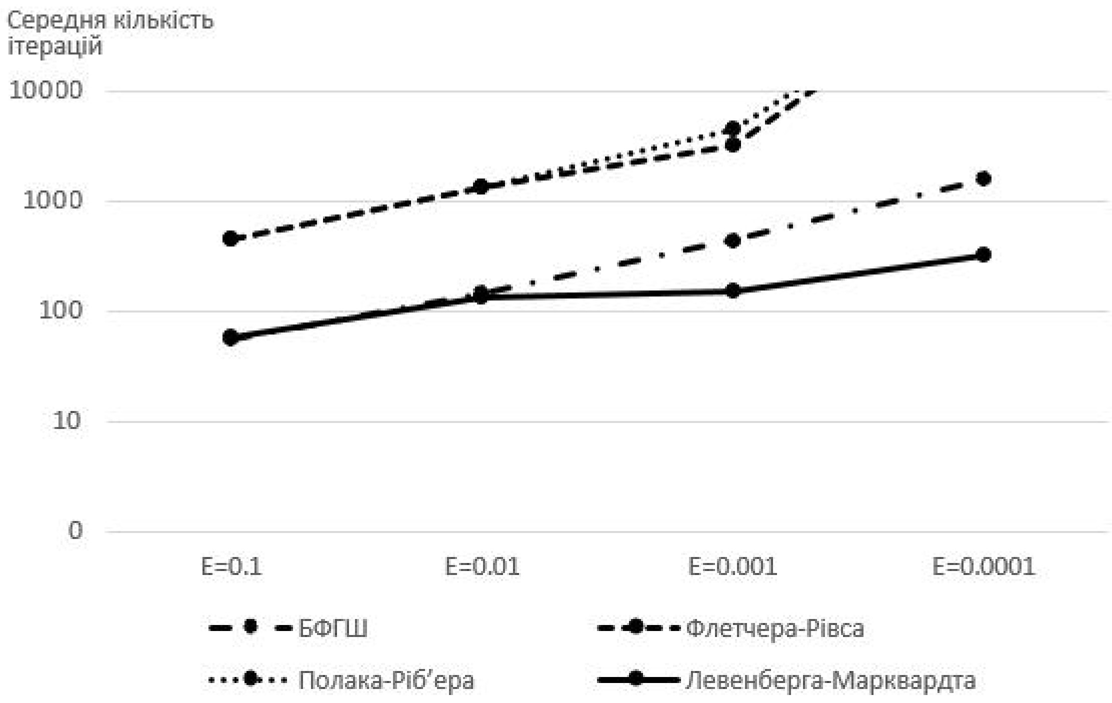

Рисунок 4.4 – Оцінка швидкості збіжності алгоритмів навчання при моделюванні функції tan(x)

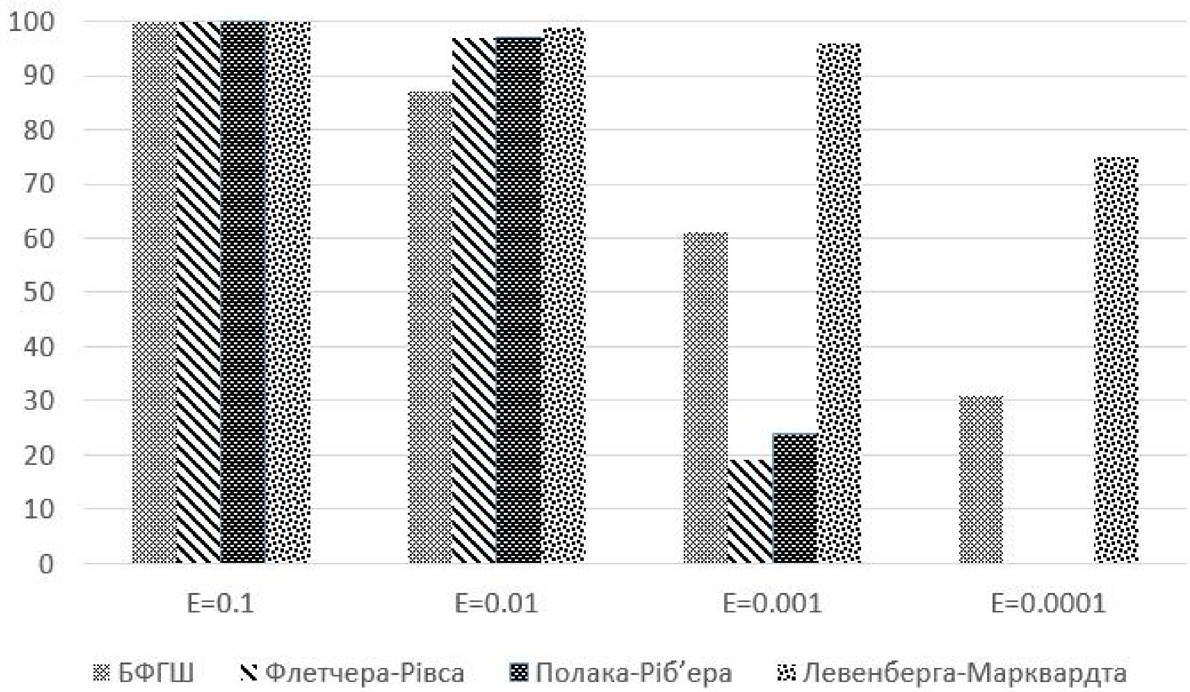

Рисунок 4.5 – Оцінка глобальної надійності алгоритмів навчання при моделюванні функції tan(x)

Таблиця 4.3 – Оцінка алгоритмів навчання при моделюванні функції sin(x)cos(2x)

Метод	E=0.1		E=0.01		E=0.001		E=0.0001	
	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС
БФГШ	4.59	100	13.7	100	134	100	265	100
Флетчера-Рівса	7.65	100	36.4	100	568	92	4753	38
Полака-Ріб'ера	12.56	100	108	100	3047	74	4756	24
Левенберга-Марквардта	1.56	100	3.45	100	10.45	99	38.6	95

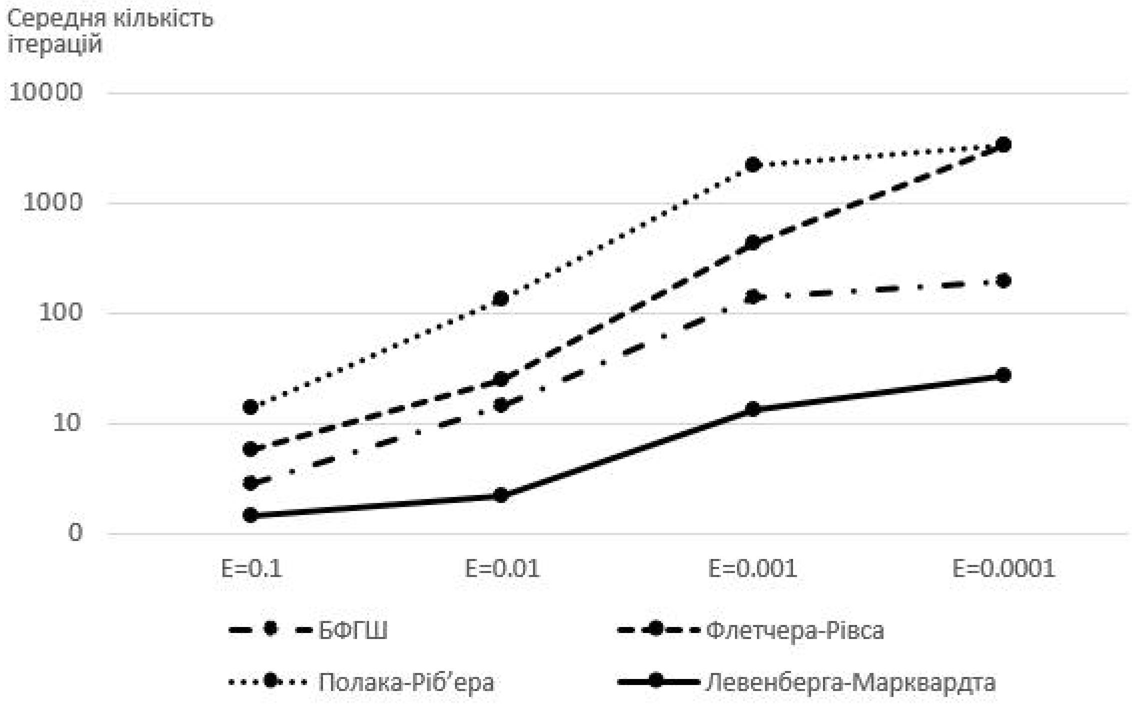

Рисунок 4.6 – Оцінка швидкості збіжності алгоритмів навчання при моделюванні функції sin(x)cos(2x)

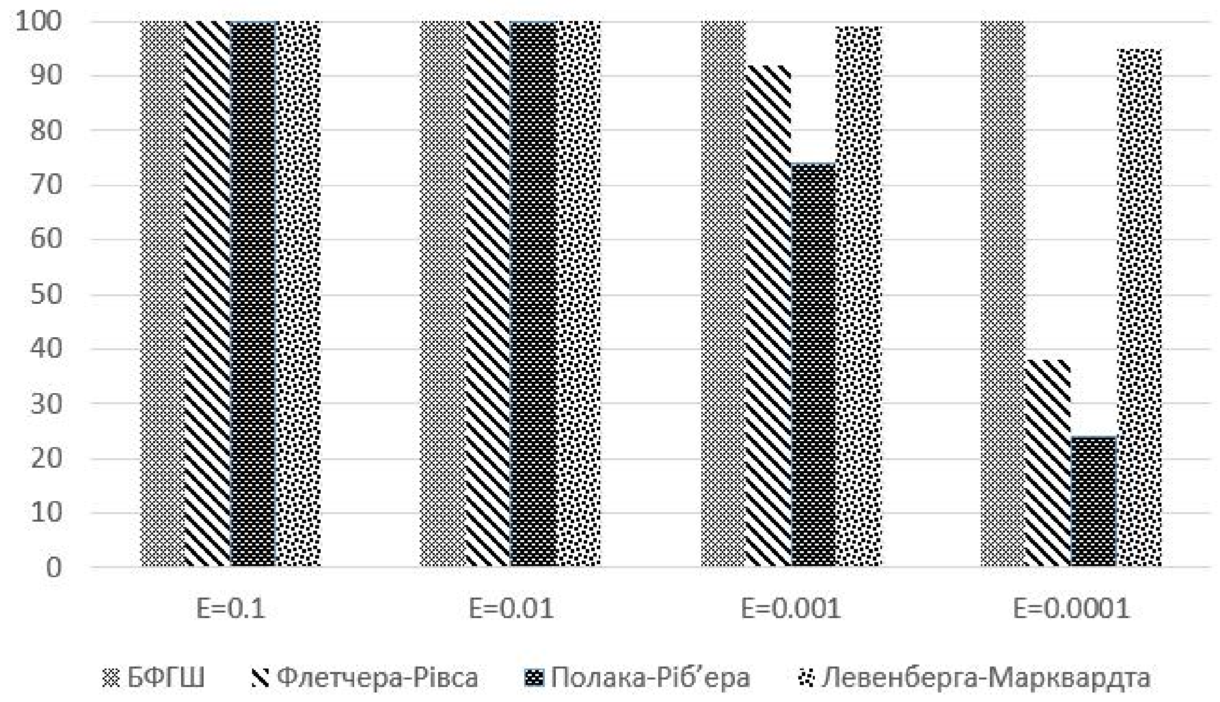

Рисунок 4.7 – Оцінка глобальної надійності алгоритмів навчання при моделюванні функції sin(x)cos(2x)

Виходячи з експериментальних даних методи оптимізації можна розташувати в наступному порядку (в порядку зменшення обчислювальної стійкості і швидкості збіжності):

- метод Левенберга-Марквардта;
- квазіньютоновський метод з модифікацією Бройдена-Флетчера-Гольдфарба-Шанно;
- метод Флетчера-Рівса;
- метод Полака-Ріб’ера.

Найшвидшим виявився метод Левенберга-Марквардта, але його не можна рекомендувати для будь-яких завдань через чутливість до помилок округлення. Ефективне виконання квазіньютоновського алгоритму з модифікацією Бройдена-Флетчера-Гольтфарба-Шанно заслуговує найвищої оцінки. Він вимагає менших обчислювальних витрат на кожній ітерації, ніж метод Ньютона, і, на відміну від методу Левенберга-Марквардта, не чутливий до помилок округлення. Крім того, теоретична швидкість збіжності даного методу вище, ніж у методів сполучених градієнтів. На жаль, високі витрати, необхідні для зберігання даних, не дозволяють використовувати квазіньютоновські алгоритми для вирішення задач великої розмірності. Далі йдуть методи спряжених градієнтів, які можна рекомендувати для вирішення завдань середнього та великого ступеня складності зважаючи на простоту реалізації, невисоких обчислювальних витрат і малого об’єму необхідної пам’яті.

Що стосується складності реалізації, то розглянуті алгоритми повинні бути розташовані в порядку, зворотному до представленого вище.

4.3 Моделювання процесу ідентифікації

4.3.1 Модифікація методу Левенберга-Марквардта для РБМ

Розглянемо застосування для навчання мережі методу другого порядку – методу Левенберга-Марквардта, що є реалізацією відомого методу безумовної оптимізації [12]. Метод Левенберга-Марквардта широко застосовується для навчання багатошарових персептронів [23], але практично не застосовується для навчання мереж радіальних базисних функцій. Слід відзначити роботу [30], в якій запропонована ефективна з обчислювальної точки зору апроксимація матриці Гессе, розглянута реалізація методу Гауса-Ньютона для навчання РБФ-мереж, але не розглянута реалізація методу Левенберга-Марквардта.

Розглянемо налаштування параметрів. Введемо єдиний вектор параметрів:

$$\theta=[\omega_1,\omega_2,...,\omega_n,c_{11},c_{21},...,c_{n1},c_{12},c_{22},...,c_{n2},\sigma_1,\sigma_2,...\sigma_n]^T, \qquad (4.1)$$

де j-й елемент – параметр РБ-функції (j=1, 2, 3,…,n);

ω_j – вага;

c_{j1} та c_{j2} – координати центру (розглядаємо апроксимацію функцій двох змінних);

σ_j – відхилення.

Налаштування вектору параметрів θ в k-ій ітерації здійснюється за формулою:

$$\theta(k) = \theta(k-1) + \Delta\theta(k)\,. \tag{4.2}$$

Де вектор виправлення $\Delta\theta(k)$ знаходиться з рішення системи лінійних алгебраїчних рівнянь:

$$\left(J_{k-1}^{T} J_{k-1} + \mu_k E\right)\Delta\theta(k) = -g_{k-1}\,, \tag{4.3}$$

де J_{k-1}– матриця Якобі, обчислена за значеннями параметрів мережі в k-1 ітерації;

μ_k – параметр регуляризації, що змінюється на кожному кроці навчання;

E – одинична матриця;

$g = J^T e$ – вектор градієнта функціоналу (3.14) за вектору параметрів θ;

$e = [e_1, e_2, ..., e_n]^T$ – вектор помилок.

Матриця Якобі має вигляд:

$$J = \begin{bmatrix}
\frac{\partial e_1}{\partial \omega_1} & \cdots & \frac{\partial e_1}{\partial \omega_n} & \frac{\partial e_1}{\partial c_{11}} & \cdots & \frac{\partial e_1}{\partial c_{n1}} & \frac{\partial e_1}{\partial c_{12}} & \cdots & \frac{\partial e_1}{\partial c_{n2}} & \frac{\partial e_1}{\partial \sigma_1} & \cdots & \frac{\partial e_1}{\partial \sigma_n} \\
\frac{\partial e_2}{\partial \omega_1} & \cdots & \frac{\partial e_2}{\partial \omega_n} & \frac{\partial e_2}{\partial c_{11}} & \cdots & \frac{\partial e_2}{\partial c_{n1}} & \frac{\partial e_2}{\partial c_{12}} & \cdots & \frac{\partial e_2}{\partial c_{n2}} & \frac{\partial e_2}{\partial \sigma_1} & \cdots & \frac{\partial e_2}{\partial \sigma_n} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\frac{\partial e_n}{\partial \omega_1} & \cdots & \frac{\partial e_n}{\partial \omega_n} & \frac{\partial e_n}{\partial c_{11}} & \cdots & \frac{\partial c_{n1}}{\partial c_{n1}} & \frac{\partial e_n}{\partial c_{12}} & \cdots & \frac{\partial c_{n2}}{\partial c_{n2}} & \frac{\partial e_n}{\partial \sigma_1} & \cdots & \frac{\partial e_n}{\partial \sigma_n}
\end{bmatrix}. \tag{4.4}$$

Наведемо матрицю Якобі (4.4) в блоковому вигляді:

$$J=\begin{bmatrix} J_{\omega} & J_{c1} & J_{c2} & J_{\sigma} \end{bmatrix}. \qquad (4.4)$$

Елементи матриці J_{ω} з урахуванням (3.1) і (3.14) мають вигляд:

$$\frac{\partial e_i}{\partial \omega_j}=\frac{\partial}{\partial \omega_j}\left[f(x_i)-\mathrm{int}(x_i)\right]=\frac{\partial f(x_i)}{\partial \omega_j}=\varphi_j(x_i), \qquad (4.5)$$

де $\varphi_j(x_i)$ – значення j-й радіальної базисної функції (2) в навчальній точці x_i.

Елементи матриці J_{c1} описуються формулою:

$$\frac{\partial e_i}{\partial c_{j1}}=\frac{\partial}{\partial c_{j1}}\left[f(x_i)-\mathrm{int}(x_i)\right]=\frac{\partial}{\partial c_{j1}}\left[\sum_{k=1}^{n}\omega_k\varphi_k(x_i)\right]=\omega_i\frac{\partial}{\partial c_{j1}}\left[\frac{(p_{j1}-c_{j1})^2+(p_{j2}-c_{j2})^2}{2\sigma_j^2}\right]=$$
$$=\omega_i e^{\frac{\|p_i-c_j\|^2}{2\sigma_j^2}}\cdot\frac{\partial}{\partial c_{j1}}\left[-\frac{(p_{j1}-c_{j1})^2+(p_{j2}-c_{j2})^2}{2\sigma_j^2}\right]=\omega_i\varphi_j(x_i)\frac{p_{i1}-c_{j1}}{\sigma_j^2}. \qquad (4.6)$$

Аналогічно для елементів матриці J_{c2} отримуємо:

$$\frac{\partial e_i}{\partial c_{j2}}=\omega_i\varphi_j(x_i)\frac{p_{i2}-c_{j2}}{\sigma_j^2}. \qquad (4.7)$$

Елементи матриці J_{σ} вираховуються формулою:

$$\frac{\partial e_i}{\partial \sigma_j}=\frac{\partial}{\partial \sigma_j}\left[f(x_i)-\operatorname{int}(x_i)\right]=\frac{\partial}{\partial \sigma_j}\left[\sum_{k=1}^{n}\omega_k\varphi_k(x_i)\right]=\omega_i\frac{\partial}{\partial \sigma_j}\left[e^{-\frac{\|p_i-c_j\|^2}{2\sigma_j^2}}\right]=$$
$$=\omega_i e^{-\frac{\|p_i-c_j\|^2}{2\sigma_j^2}}\frac{\partial}{\partial \sigma_j}\left[-\frac{\|p_i-c_j\|^2}{2\sigma_j^2}\right]=\omega_i\varphi_j(x_i)\frac{\|p_i-c_j\|^2}{\sigma_j^3}. \qquad (4.8)$$

Матриця $J_{k-1}^T J_{k-1}+\mu_k E$ є щільною симетричною і позитивно визначеною. Тому для вирішення системи (4.3) доцільно використовувати метод Холецького [30], реалізований в математичному пакеті MatLab.

4.3.2 Практична реалізація модифікації методу Левенберга-Марквардта

Розглядалася задача ідентифікації нелінійного динамічного об’єкта, що описується рівнянням [18-20]:

$$y(k)=0.725\sin\left(\frac{16u(k-1)+8y(k-1)}{(3+4u^2(k-1)+4y^2(k-1))}\right)+2u(k-1)+0.2y(k-1), \qquad (4.9)$$

де $u(k)$ – змінюваний у часі параметр.

Вхідний сигнал $u(k)$ є випадковою стаціонарною послідовністю з рівномірним законом розподілу в інтервалі $[-1,1]$. Навчальний набір включає 5000 навчальних пар. Необхідна точність задавалася на рівні $\varepsilon=0.0001$. Кількість нейронів – 20. При навчанні РБМ використовувався модифікований алгоритм Левенберга-Марквардта розглянутий у попередньому пункті.

Результати ідентифікації об’єкта (4.1) представлені на рис. 4.8. Тут показані сама поверхня, що описується заданим рівнянням, та її перерізи. Заданої точності рішення РБМ досягла за 42 ітерації за час 10,2 хвилини.

Посилення обчислювальної потужності РБМ завдяки підвищенню кількості нейронів призводило до скорочення числа ітерацій, необхідних для досягнення заданої точності, але при цьому істотно зросла час навчання, яке в даному випадку склало 15,5 хвилин.

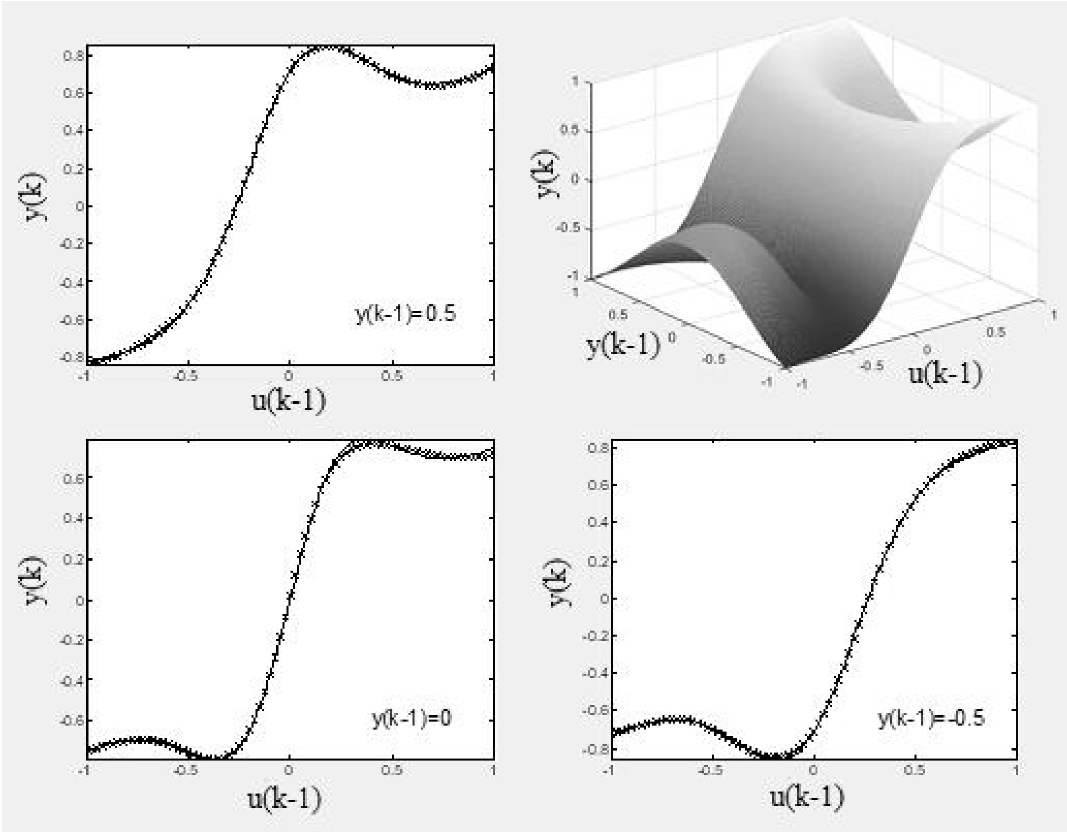

Рисунок 4.8 – Результати ідентифікації нелінійного динамічного об’єкта (4.9)

Представлені результати ідентифікації об’єкта (4.9) при наявності на його виході випадкової перешкоди, що має нормальний закон розподілу в інтервалі $[-0.25, 0.25]$ (рисунок 4.9).

Для вирішення цього завдання використовувалася структура мережі з 20 нейронами. Як видно з рисунка, мережа успішно впоралася з поставленим завданням, завдяки своїм високим фільтруючим властивостям. На перетинах суцільний жирною лінією показана реакція об’єкта, а кружечками – вихід мережі при відсутності перешкоди, реакція об’єкта і мережі при наявності шуму показані суцільною тонкою лінією і хрестиками відповідно. Час навчання склало 13,4 хвилини.

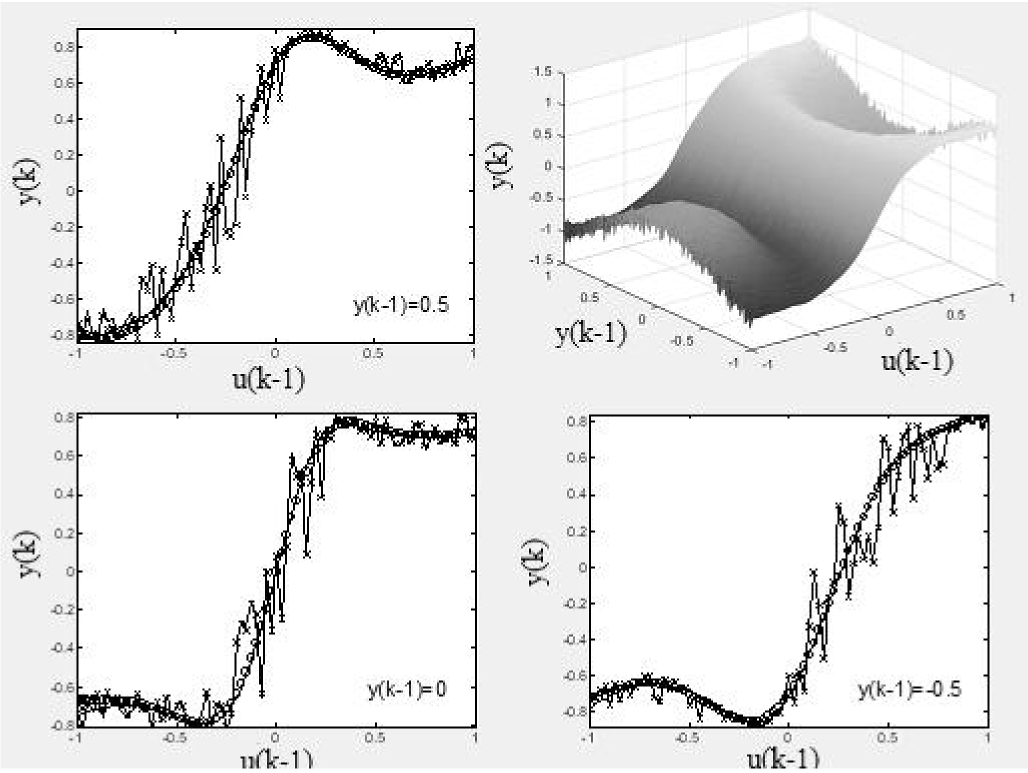

Рисунок 4.9 – Результати ідентифікації нелінійного динамічного об’єкта (4.9) при наявності випадкової перешкоди

Як показує практика, використання ШНМ, зокрема РБМ, ефективно в тих випадках, коли побудова математичної моделі реального об’єкта традиційними методами є досить проблематичним. Однак якість розв’язання задачі ідентифікації нелінійних динамічних об’єктів за допомогою РБМ в свою чергу в значній мірі залежить від вибору алгоритму його навчання. Використання алгоритму Левенберга-Марквардта, що володіє хорошими фільтруючими і згладжуючими властивостями і тому забезпечує отримання задовільних результатів при наявності перешкод вимірів.

ВИСНОВКИ

У ході виконання атестаційної роботи було проведено аналіз існуючих методів опису та ідентифікації нелінійних динамічних об’єктів. Проаналізовані принципи побудови штучних нейронних мереж. ШНМ добре підходять для задачі ідентифікації нелінійних динамічних об’єктів завдяки своїй адаптивності, спроможності до самонавчання, високій швидкодії. На підставі проведеної роботи була побудована модель ідентифікації нелінійних динамічних об’єктів на основі радіально-базисної мережі.

Проведене імітаційне моделювання наступних алгоритмів навчання радиально-базисної мережі:

- метод Левенберга-Марквардта;
- квазіньютоновський метод з модифікацією Бройдена-Флетчера-Гольдфарба-Шанно;
- метод Флетчера-Рівса;
- метод Полака-Ріб’ера.

За результатами імітаційного моделювання цих алгоритмів навчання РБМ виконаний аналіз їх надійності та швидкості збіжності, на підставі чого розроблено рекомендації щодо їх практичного використання.

Досліджено процес вирішення завдань ідентифікації нелінійними динамічними об’єктами при впливі перешкод. Показана ефективність використання алгоритму Левенберга-Марквардта, що володіє хорошими фільтруючими і згладжуючими властивостями і тому забезпечує отримання задовільних результатів при наявності перешкод вимірів.

Надалі можна розширити дослідження провевши порівняльний аналіз використання різних функцій активації РБМ.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Nelles, O. Nonlinear System Identification [Text] / O. Nelles. – Berlin: Springer. – 2001. – 785 p;

2. Медведев, А.В. Непараметрические алгоритмы идентификации нелинейных динамических систем [Текст] / А.В. Медведев // Стохастические системы управления. – Новосибирск: Наука. – 2001. – С. 15-22.

3. Перельман, И.И. Оперативная идентификация объектов управления. [Текст] / И.И. Перельман – М.: Энергоиздат, 1982. – 272с.

4. Эйкхофф, П. Основы идентификации систем управления [Текст] : пер. с англ. – М.: Мир, 1975. – 683с.

5. Suykens, J.A.K. Artifical Neural Networks for Modelling and Control of Non-Linear Systems [Text] / J.A.K. Suykens, J.P.L. Vandewille, B.L.R. De Moor– Boston: Kluwer Academic Publishers, 1996. – 235p.

6. Plett, G.L. Adaptive inverse control of linear/nonlinear systems using dynamic neural networks [Text] / G.L. Plett // IEEE Trans. Neural Networks. – 2003. – Vol.5. – N2. – P.360-376.

7. Козынченко, В.А. Нейронная сеть в задачах апроксимации [Текст] / В.А. Козынченко, А.И. Прус, // IEEE Publications, – Санкт-Петербург, 2014. – С.23–27.

8. Хайкин, С. Нейронные сети. Полный курс [Текст] : учеб. пособие / С. Хайкин – М.: Издательский дом Вильямс, 2006. – 1104 с.

9. Булатов, А.Ф. Выбор оптимальной архитектуры нейронной сети для задач диагностики [Текст] / А.Ф. Булатов // Наукоёмкие технологии в машиностроении: материалы Всероссийской НПК (Ишимбай, 17–18 мая 2013) – Уфа: УГАТУ, 2013. – С.32–33.

10. Головко, В.А. Нейронные сети: обучения, организация и применение [Текст] / В.А. Головко – М.: ИПРЖР. 2008. – 435с.

11. Иванов, А.И. Нейросетевые алгоритмы биометрической

идентификации [Текст] / А.И. Иванов – М.: Издво: Радиотехника. 2006. – 144 с.

12. Руденко, О.Г. Исскуственные нейронные сети [Текст] / О.Г. Руденко, Е.В. Бодянский – Харьков: ООО «Компания СМИТ», 2005. – 408 с.

13. Каллан, Р. Основные концепции нейронных сетей [Текст] / Р. Каллан – М.: Изд-во: Вильямс. 2002. – 287 с

14. Крисилов, В.А. Представление исходных данных в задачах нейросетевого программирования [Текст] / В.А. Крисилов– Одесса: ОНПУ. – 2003с.

15. Леван, Д.Н. Особенности использования многослойного персептрона [Текст] / Д.Н. Леван, Н.А. Феоктистов – К.: Науковедени. вып. 2. – 2014. – С. 34–36.

16. Солдатова, О.П., Применение нейронных сетей для решения задач прогнозирования [Текст] / О.П. Солдатова, В.В. Семенов – Саратов: СГАУ им. ак. С.П. Королева, 2010. – С. 56-58.

17. Park, J. Universal approximation using radial-basis-function networks [Text] / J. Park, I.W. Sandberg // Neural Computation. – 1991. – 3. – P.246-257.

18. Schilling, R.J. Approximation of nonlinear systems with radial basis function neural networks [Text] / R.J. Schilling, J.J. Carrol, A.F. Al-Ajlouni // IEEE Trans. on Neural Networks. – 2001. – P. 1-15.

19. Kowaliw, T. Artificial Neurogenesis: An Introduction and Selective Review[Text] / T. Kowaliw, N. Bredeche, S. Chevallier, R. Doursat // Studies in Computational Intelligence. – 2014. – P.1-60

20. Kruse, R. Computational Intelligence. A Methodological Introduction [Text] / R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, M. Steinbrecher, P. Held, P – Berlin: Springer, 2013. – 488 p.

21. Du, K.-L. Neural Networks and Statistical Learning [Text] / K.-L. Du, M.N.S. Swamy – London: Springer-Verlag, 2014. – 824 p.

22. Jin, Y. Computational Intelligence in Big Data [Text] / Y. Jin, B. Hammer // IEEE Computational Intelligence Magazine. – 2014. – August. – P. 12-15.

23. Pedrycz, Eds. W. Information Granularity, Big Data, and Computational

Intelligence [Text] / Eds. W. Pedrycz, Shyi-Ming Chen. – Cham: Springer, 2015. – 444p.

24. Wang, L. X. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning [Text] / L. X. Wang, J. M. Mendel // IEEE Transactions on Neural Networks – Vol. 3, Issue 5. 1992. – P. 807-814.

25. Kacprzyk, J. Springer Handbook of Computational Intelligence [Text] / J. Kacprzyk, W. Pedrycz. – Berlin Heidelberg: Springer – Verlag. – 2015. – 1634 p

26. Bodyanskiy, Ye. Nonlinear process identification and modeling using generalized regression neuro-fuzzy network [Text] / Ye. Bodyanskiy , P. Otto, I. Pliss, N. Teslenko // Proc. 52 nd Int. Sci. Coll. «Computer Science Meets Automation». – TU Ilmenau (Thuer). – 2007. – P.27-33.

27. Нечаев, Ю.И. Нейроапроксимация и нейропрогноз при контроле динамики сложного объекта [Текст] / Ю.И. Нечаев // Нейрокомпьютеры: разработка, применение. – 2005. – № 10-11. – С. 22-31.

28. Горбань, А.Н. Нейронные сети на персональном компьютере. [Текст] / А.Н. Горбань, Д.А. Россиев – Новосибирск: Наука, 1996. – 276 с.

29. Дли, М.И. Нечеткая логика и искусственные нейронные сети [Текст] / М.И. Дли– М.: Физматлит, 2001. – 225 с.

30. Дубровін, В.І. Субботін С.О. Методи оптимізації та їх застосування в задачах навчання нейронних мереж [Текст] : Навчальний посібник / В.І. Дубровін, С.О. Субботін – Запоріжжя: ЗНТУ, 2003. – 136 с.

31. Каллан, Р. Основные концепции нейронных сетей [Текст] / Р. Каллан,– М.: Издательский дом «Вильямс», 2001. – 287 с.

32. Круглов, В.В. Искусственные нейронные сети: теория и практика [Текст] / В.В. Круглов, В.В. Борисов – М.: Горячая линия – Телеком, 2001. – 382 с.

33. Demuth, H. Neural Network Toolbox for use with MATLAB: user’s guide [Text] / H. Demuth, M. Beale. – Natick: Mathworks Inc, 1997. – 700 p.