Journal of Applied Mathematics

Qualitative Analysis of Dynamic Activity Patterns in Neural Networks

Guest Editors: Ivanka Stamova, Haydar Akca, and Gani Stamov

Qualitative Analysis of Dynamic Activity Patterns in Neural Networks

Journal of Applied Mathematics

Qualitative Analysis of Dynamic Activity Patterns in Neural Networks

Guest Editors: Ivanka Stamova, Haydar Akca, and Gani Stamov

Editorial Board

Contents

Qualitative Analysis of Dynamic Activity Patterns in Neural Networks, Ivanka Stamova, Haydar Akca, and Gani Stamov
Volume 2011, Article ID 208517, 2 pages

Pseudo Almost-Periodic Solution of Shunting Inhibitory Cellular Neural Networks with Delay, Haihui Wu
Volume 2011, Article ID 510789, 14 pages

Encoding Static and Temporal Patterns with a Bidirectional Heteroassociative Memory, Sylvain Chartier and Mounir Boukadoum
Volume 2011, Article ID 301204, 34 pages

Hysteresis Nonlinearity Identification Using New Preisach Model-Based Artificial Neural Network Approach, Mohammad Reza Zakerzadeh, Mohsen Firouzi, Hassan Sayyaadi, and Saeed Bagheri Shouraki
Volume 2011, Article ID 458768, 22 pages

An ANFIS Approach for Real Power Transfer Allocation, Hussain Shareef, Saifulnizam Abd Khalid, Mohd Wazir Mustafa, and Azhar Khairuddin
Volume 2011, Article ID 414258, 14 pages

Downscaling Global Weather Forecast Outputs Using ANN for Flood Prediction, Nam Do Hoai, Keiko Udo, and Akira Mano
Volume 2011, Article ID 246286, 14 pages

Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2011, Article ID 208517, 2 pages
doi:10.1155/2011/208517

Editorial

Qualitative Analysis of Dynamic Activity Patterns in Neural Networks

Ivanka Stamova,[1] Haydar Akca,[2] and Gani Stamov[3]

[1] *Center of Computer Science and Engineering, Burgas Free University, 8000 Burgas, Bulgaria*
[2] *Department of Applied Sciences and Mathematics, College of Arts and Sciences, Abu Dhabi University, P.O. Box 59911 Abu Dhabi, UAE*
[3] *Department of Mathematics, Technical University of Sofia, 8800 Sliven, Bulgaria*

Correspondence should be addressed to Ivanka Stamova, stamova@bfu.bg

Received 3 May 2011; Accepted 3 May 2011

Neural networks have recently been widely used to model some of the human activities in many areas of science and engineering. Mathematical modeling in neural networks has been based on "neurons" that is different both from real biological neurons and from the realistic functioning of simple electronic circuits. These models have received increasing interest due to their impressive applications in areas such as classification, parallel computing, associative memory, pattern recognition, computer vision, and solving some optimization problem. Such applications heavily depend on the dynamic behavior of networks; therefore, the qualitative analysis of these dynamic behaviors is a necessary step for practical design of neural networks.

Neural networks have broad applicability to real-world business problems. In fact, they have already been successfully applied in many industries. Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including sales forecasting, industrial process control, customer research, data validation, risk management, and target marketing.

In this special issue on multimedia networking, we have invited a few papers that address such issues.

In the first paper, shunting inhibitory cellular neural networks with delay are studied. By using the Lyapunov functional and contraction mapping, a set of criteria are established for the global exponential stability, the existence, and uniqueness of pseudo-almost-periodic solutions.

The second paper is on a bidirectional associative memory (BAM) model. Several interesting properties of the BAM architecture have been investigated. The analysis of the

transmission function, as well as one-to-one association and many-to-one association are studied. It was shown that simple time-delay Hebbian learning can perform one-to-one association and many-to-one association with both binary and real-valued pattern. More complex architectures are explored by using the multidirectional associative memory (MAM) architecture while keeping the same learning and transmission functions. Simulations show the BAM's various capacities by using several types of encoding and recall situations.

The third paper describes a novel hysteresis identification method based on numerical classical Preisach model by use of artificial neural networks. The experimental data showed that the new approach provides accurate hysteresis nonlinearity modeling in comparison with the classical Preisach model and can be used for many applications such as hysteresis nonlinearity control and identification in SMA and Piezo actuators and performance evaluation in some physical systems such as magnetic materials.

The fourth paper presents an adaptive neuro fuzzy interface system approach to identify the real power transfer between generators. The proposed method could be adapted to true application of real power allocation and help to resolve some of the difficult real power pricing and costing issues to ensure fairness and transparency in the deregulated environment of power system operation.

Finally, the fifth paper of this special issue is devoted to an empirical-statistical downscaling method for precipitation prediction which uses a feed-forward multilayer perceptron neural network. The downscaling model has taken into account the physical bases of the precipitation evolution induced by meteorological and land-surface characteristics in the study area. As a result, the present model has exhibited cost-effective, simple-to-implement, and universal application.

Ivanka Stamova
Haydar Akca
Gani Stamov

Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2011, Article ID 510789, 14 pages
doi:10.1155/2011/510789

Research Article

Pseudo Almost-Periodic Solution of Shunting Inhibitory Cellular Neural Networks with Delay

Haihui Wu

Sunshine College, Fuzhou University, Fuzhou, Fujian, 350002, China

Correspondence should be addressed to Haihui Wu, whh3346@sina.com

Received 28 June 2010; Accepted 29 July 2010

Academic Editor: Ivanka Stamova

Shunting inhibitory cellular neural networks are studied. Some sufficient criteria are obtained for the existence and uniqueness of pseudo almost-periodic solution of this system. Our results improve and generalize those of the previous studies. This is the first paper considering the pseudo almost-periodic SICNNs. Furthermore, several methods are applied to establish sufficient criteria for the globally exponential stability of this system. The approaches are based on constructing suitable Lyapunov functionals and the well-known Banach contraction mapping principle.

1. Introduction

It is well known that the cellular neural networks (CNNs) are widely applied in signal processing, image processing, pattern recognition, and so on. The theoretical and applied studies of CNNs have been a new focus of studies worldwide (see [1–12]). Bouzerdoum and Pinter in [1] have introduced a new class of CNNs, namely, the shunting inhibitory CNNs (SICNNs). Shunting neural networks have been extensively applied in psychophysics, speech, perception, robotics, adaptive pattern recognition, vision, and image processing. Recently, Chen and Cao [9] have studied the existence of almost-periodic solutions of the following system of SICNNs:

$$\dot{x}_{ij}(t) = -a_{ij}x_{ij}(t) - \sum_{C^{kl}\in N_r(i,j)} c_{ij}^{kl} f\left(x_{ij}(t-\tau)\right)x_{ij}(t) + L_{ij}(t), \tag{1.1}$$

where C_{ij} denotes the cell at the (i,j) position of the lattice, the r-neighborhood $N_r(i,j)$ of C_{ij} is

$$N_r(i,j) = \left\{ C_{ij}^{kl} : \max\left(|k-l|, \left|l-j\right| \le r\right), 1 \le k \le m, 1 \le l \le n \right\}, \tag{1.2}$$

x_{ij} is the activity of the cell C_{ij}, $L_{ij}(t)$ is the external input to C_{ij}, the constant $a_{ij} > 0$ represents the passive decay rate of the cell activity, $C_{ij}^{kl} \ge 0$ is the connection or coupling strength of postsynaptic activity of the cell transmitted to the cell C_{ij}, and the activation function $f(x_{kl})$ is a positive continuous function representing the output or firing rate of the cell C_{ij}. Since studies on neural dynamic systems not only involve a discussion of stability properties, but also involve many dynamic properties such as periodic oscillatory behavior, almost-periodic oscillatory properties, chaos, and bifurcation. To the best of our knowledge, few authors have studied almost-periodic solutions for SINNs with delays and variable coefficients, and most of them discuss the stability, periodic oscillation in the case of constant coefficients. In their paper, they investigated the existence and stability of periodic solutions of SINNs with delays and variable coefficients. They considered the SICNNs with delays and variable coefficients

$$\begin{aligned} \dot{x}_{ij}(t) = -a_{ij}(t)x_{ij}(t) &- \sum_{B^{kl} \in N_r(i,j)} B_{ij}^{kl}(t) f_{ij}(x_{kl}(t)) x_{ij}(t) \\ &- \sum_{C^{kl} \in N_r(i,j)} C_{ij}^{kl}(t) g_{ij}(x_{kl}(t-\tau_{kl})) x_{ij}(t) + L_{ij}(t), \end{aligned} \tag{1.3}$$

where, for each $i = 1,2,\ldots,n$, $j = 1,2,\ldots,m$, $a_{ij}(t), B_{ij}^{kl}(t), C_{ij}^{kl}(t)$,and $L_{ij}(t)$ are all continuous $\omega-$ periodic functions and $a_{ij}(t) > 0, B_{ij}^{kl}(t) \ge 0, C_{ij}^{kl}(t) \ge 0$, and τ_{ij} is a positive constant.

In this paper, we consider the following more general SICNNs:

$$\begin{aligned} \dot{x}_{ij}(t) = -a_{ij}(t)x_{ij}(t) &- \sum_{B^{kl} \in N_r(i,j)} B_{ij}^{kl}(t) f_{ij}(x_{kl}(t)) x_{ij}(t) \\ &- \sum_{C^{kl} \in N_r(i,j)} C_{ij}^{kl}(t) g_{ij}(x_{kl}(t-\tau_{kl}(t))) x_{ij}(t) + L_{ij}(t). \end{aligned} \tag{1.4}$$

By using the Lyapunov functional and contraction mapping, a set of criteria are established for the globally exponential stability, the existence, and uniqueness of pseudo almost-periodic solution for the SICNNs. This is the first paper considering the pseudo almost-periodic solution of SICNNs. Since the nature is full of all kinds of tiny perturbations, either the periodicity assumption or the almost-periodicity assumption is just approximation of some degree of the natural perturbations. A well-known extension of almost periodicity is the asymptotically almost periodicity, which was introduced by Frechet. In 1992, Zhang [13, 14] introduced a more general extension of the concept of asymptotically almost periodicity, the so-called pseudo almost periodicity, which has been widely applied in the theory of ODEs and PDEs. However, it is rarely applied in the theory of neural networks or mathematical biology. This paper is expected to establish criteria that provide much flexibility

in the designing and training of neural networks and to shed some new light on the application of pseudo almost periodicity in neural networks, population dynamics, and the theory of differential equations.

Throughout this paper, we will use the notations $g^M = \sup_{t\in R} g(t), g^L = \sup_{t\in R} g(t)$, where $g(t)$ is a bounded continuous function on R.

In this paper, we always use $i = 1, 2, \ldots, m; j = 1, 2, \ldots, n$; unless otherwise stated.

In this paper, we always consider system (1.4) together with the following assumptions.

- (A_1) $a_{ij}(t)$ are almost periodic on R with $a_{ij}(t) > 0$, and $L_{ij}(t)$ and $\tau_{ij}(t)$ are pseudo almost periodic on R with $L_{ij} > 0$.
- (A_2) $\tau_{ij}(t)$ is bounded, continuous and differentiable with $0 \le \tau_{ij} \le \tau, 1 - \tau > 0$ for $t \in R$, where τ is constant.
- (A_3) $f_{ij}, g_{ij} \in C(R, R)$ are bounded, and continuous, and there exist positive numbers μ_{ij}, ν_{ij} such that $|f_{ij}(x) - f_{ij}(y)| \le \mu_{ij}|x - y|, |g_{ij}(x) - g_{ij}(y)| \le \nu_{ij}|x - y|$, for all $x, y \in R$.

2. Preliminaries and Basic Results of Pseudo Almost-Periodic Function

In this section, we explore the existence of pseudo almost-periodic solution of (1.4). First, we would like to recall some basic notations and results of almost periodicity and pseudo almost periodicity [15, 16] which will come into play later on.

Let $\Omega \subset C^n$ be close and let $\mathcal{L}(R)$ (resp., $\mathcal{L}(R \times \Omega)$) denote the C^*-algebra of bounded continuous complex-valued functions on R (respectively, $R \times \Omega$) with supremum norm $|\cdot|_\infty$ denoting the Euclidean norm in C^n; that is, $|x|_\infty = \max_{1\le i\le n}|x_i|$, $x \in C^n$.

Definition 2.1. A function $g \in \mathcal{L}(R)$ is called almost periodic if, for each $\epsilon > 0$, there exists an $l_\epsilon > 0$ such that every interval of length l_ϵ contains a number τ with the property that $|g(t + \tau) - g(t)| < \epsilon, t \in R$.

Definition 2.2. A function $g \in \mathcal{L}(R \times \Omega)$ is called almost periodic in $t \in R$, uniformly in $Z \in \Omega$, if, for each $\epsilon > 0$ and any compact set M of Ω, there exists an $l_\epsilon > 0$ such that every interval of length l_ϵ contains a number τ with the property that $|g(t + \tau, z) - g(t, z)| < \epsilon, t \in R, Z \in \Omega$. The number τ is called an ϵ-translation number of g. Denote by $\mathcal{AP}(R)(\mathcal{AP}(R \times \Omega))$ the set of all such function.

Set

$$\mathcal{PAP}_0(R) = \left\{\varphi \in \mathcal{L}(R) : \lim_{t\to\infty} \frac{1}{2t}\int_{-t}^{t} |\varphi(s)|ds = 0\right\},$$
$$\mathcal{PAP}_0(R \times \Omega) = \left\{\varphi \in \mathcal{L}(R \times \Omega) : \lim_{t\to\infty} \frac{1}{2t}\int_{-t}^{t} |\varphi(s, Z)|ds = 0, \text{uniformly in } Z \in \Omega\right\}. \tag{2.1}$$

Definition 2.3. A function $f \in \mathcal{L}(R)(\mathcal{L}(R \times \Omega))$ is called pseudo almost periodic (pseudo almost periodic in $t \in R$, uniformly in $Z \in \Omega$), if $f = g + \varphi$, where $g \in \mathcal{PA}(R)(\mathcal{PA}(R \times \Omega))$ and $\varphi \in \mathcal{PAP}_0(R)(\mathcal{PAP}_0(R \times \Omega))$. The function g and φ are called the almost periodic component

and the erigodic perturbation, respectively, of the function f. Denote by $\mathcal{PAP}(R)(\mathcal{PAP}(R \times \Omega))$ the set of all such functions f.

Define $\|x\|_\infty = \sup_{t\in R}|x(t)|_\infty, x \in \mathcal{PAP}$. It is trivial to show that $\mathcal{PAP}$ is a Banach space with $\|\cdot\|_\infty$.

Let $A(t) = (a_{ij}(t))$ be a complex $n \times n$ matrix-valued function with elements (entries) which are continuous on R. We consider the homogeneous linear ODE and nonhomogeneous linear ODE as follows:

$$\frac{dx}{dt} = A(t)x, \tag{2.2}$$

$$\frac{dx}{dt} = A(t)x + f(t), \tag{2.3}$$

where x denotes an n-column vector.

Definition 2.4 (see [15, 16]). The homogeneous linear ODE (2.2) is said to admit an exponential dichotomy if there exist a linear projection p (i.e., $p^2 = P$) on C^n and positive constants k, α, β such that

$$\begin{aligned} \left\|X(t)PX^{-1}(s)\right\| &\le ke^{-\alpha(t-s)}, \quad t \ge s, \\ \left\|X(t)(I-P)X^{-1}(s)\right\| &\le ke^{-\beta(s-t)}, \quad t \le s, \end{aligned} \tag{2.4}$$

where $X(t)$ is a fundamental matrix of (2.2) with $X(0) = E$; E is the $n \times n$ identity matrix.

Definition 2.5 (see [15, 16]). The matrix $A(t)$ is said to be row dominant if there exists a number $\delta > 0$ such that $|\operatorname{Re} a_{ij}(t)| \ge \sum_{j=1, j\ne i}^{n} |a_{ij}(t)| + \delta$ for all $t \in (-\infty, \infty)$ and $i = 1, 2, \ldots, n$.

Lemma 2.6 (see [15, 16]). *If $A(t)$ is a bounded, continuous, and row-dominant $n\times n$ matrix function on R, and there exists $k \le n$ such that $\operatorname{Re} a_{ij} < 0$ $(i = 1, 2, \ldots, k)$. Then (2.2) has a fundamental matrix solution $X(t)$ satisfying*

$$\begin{aligned} \left\|X(t)PX^{-1}(s)\right\| &\le ke^{-\delta(t-s)}, \quad t \ge s, \\ \left\|X(t)(I-P)X^{-1}(s)\right\| &\le ke^{-\delta(s-t)}, \quad t \le s, \end{aligned} \tag{2.5}$$

where K is a positive constant, and $P = \operatorname{diag}(E_k, 0)$ with E_k being a $k \times k$ identity matrix.

For $H = (h_1, h_2, \ldots, h_n) \in \mathcal{L}(R)^n$, suppose that $H(t) \in \Omega$ for all $t \in R$. Define $H \times l : R \to \Omega \times R$ by $H \times l(t) = (h_1(t), h_2(t), \ldots, h_n(t), t)$ $(t \in R)$.

Lemma 2.7 (see [15, 16]). *Assume that the function $f(t,z) \in \mathcal{PAP}(R \times \Omega)$ is continuous in $z \in M$ uniformly in $t \in R$ for all compact subsets $M \subset \Omega$ and $F \in \mathcal{PAP}(R)^n$ such that $F(R) \subset \Omega$. Then $f \circ (F \times l) \in \mathcal{PAP}(R)$.*

It is obvious that if f satisfies a Lipschitz condition; that is, there is an $L > 0$ such that

$$\left|f(z',t) - f(z,t)\right| \le L\left|z' - z\right| \quad (z', z \in M, t \in R), \tag{2.6}$$

then f is continuous in $z \in M$ uniformly in $t \in R$. Obviously, if $f(t) \in \mathcal{PAP}(R)$ is uniformly continuous in $t \in R$ and $\varphi \in \mathcal{PAP}(R)$ such that $\varphi(R) \subset \operatorname{Im} f$, then $f \circ \varphi \in \mathcal{PAP}(R)$.

Lemma 2.8 (see [15, 16]). *Assume that $A(t)$ is an almost-periodic matrix function and $f(t) \in \mathcal{PAP}(R^n)$. If* (2.2) *satisfies an exponential dichotomy, then* (2.3) *has unique pseudo almost periodic solution $x(t)$ reading*

$$x(t) = \int_{-\infty}^{t} X(t)PX^{-1}(s)f(s)ds - \int_{t}^{\infty} X(t)(E-P)X^{-1}(s)f(s)ds \tag{2.7}$$

and satisfying $\|x\| \le (K/\alpha + K/\beta)\|f\|$, where $X(t)$ is a fundamental matrix solution of (2.2).

Definition 2.9. System (1.4) is said to be globally exponentially stable (GES), if for any two solutions $x(t)$ and $y(t)$ of (1.4), there exist positive numbers M and ε such that

$$\left|x(t) - y(t)\right|_p \le Me^{-\varepsilon(t-t_0)}\left\|\varphi - \psi\right\|_p, \quad t > t_0, \tag{2.8}$$

where $x(t) = x(t,\varphi)$ and $y(t) = y(t,\psi)$ denoting the solution of (1.4) through (t_0,φ) and (t_0,ψ) respectively. Here ε is called the Lyapunov exponent of (1.4).

3. Existence and Stability of Pseudo Almost-Periodic Solution

Theorem 3.1. *Assume that (A_1)–(A_3) hold and*

(A_4)

$$r = \sup_{t\in R}\max_{(i,j)}\left\{\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta}\left(\sum_{B^{hl}\in N_r(i,j)}\mu_{ij}B_{ij}^{hl}(s) + \sum_{C^{hl}\in N_r(i,j)}\nu_{ij}C_{ij}^{hl}(s)\right)ds\right\} < 1. \tag{3.1}$$

Then (1.4) *has a unique pseudo almost-periodic solution, say $x^*(t)$, satisfying $\|x^*\|_\infty \le L/(1-r)$.*

Proof. For any $\varphi \in \mathcal{PAP}(R)$, consider

$$\begin{aligned}\dot{x}_{ij} = -a_{ij}(t)x_{ij}(t) &- \sum_{B^{hl}\in N_r(ij)} B_{ij}^{hl}(t) f_{ij}\big(\varphi_{hl}(t)\big)\varphi_{ij}(t) \\ &- \sum_{C^{hl}\in N_r(ij)} C_{ij}^{hl}(t) g_{ij}\big(\varphi_{hl}(t-\tau_{hl}(t))\big)\varphi_{ij}(t) + L_{ij}(t).\end{aligned} \tag{3.2}$$

Since $-a_{ij}(t) < 0$, from Lemmas 2.6, 2.7, and 2.8, it follows that (3.2) has a unique pseudo almost-periodic solution, which is given by

$$\begin{aligned}x_\varphi(t) = \Bigg(\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta} \Bigg[&- \sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(s) f_{ij}\big(\varphi_{hl}(s)\big)\varphi_{ij}(s) \\ &- \sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(s) g_{ij}\big(\varphi_{hl}(s-\tau_{hl}(s))\big)\varphi_{ij}(s) + L_{ij}(s)\Bigg] ds\Bigg)_{mn\times 1}.\end{aligned} \tag{3.3}$$

Define the mapping $\mathcal{T} : \mathcal{PAP}(R) \to \mathcal{PAP}(R)$ by $\mathcal{T}(\varphi)(t) = x_\varphi(t), \varphi \in \mathcal{PAP}(R)$.

Let $B^* = \{\varphi \mid \varphi \in \mathcal{PAP}(R), \|\varphi - \varphi_0\|_\infty \le (\gamma/(1-\gamma))L\}$, where $\varphi_0(t) = (\int_{-\infty}^t e^{-\int_s^t a_{11}(\eta)d\eta} L_{11}(s)ds, \ldots, \int_{-\infty}^t e^{-\int_s^t a_{ij}(\eta)d\eta} L_{ij}(s)ds, \ldots, \int_{-\infty}^t e^{-\int_s^t a_{mn}(\eta)d\eta} L_{mn}(s)ds)^T$.

Clearly, B^* is closed and convex in B. Note that

$$\begin{aligned}\|\varphi_0\|_\infty &= \sup_{t\in R}\max_{(i,j)}\left\{\left|\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta} L_{ij}(s)ds\right|\right\} \\ &\le \sup_{t\in R}\max_{(i,j)}\left\{\left|\int_{-\infty}^{t} e^{-a_{ij}^-(s-t)} L_{ij}^+ ds\right|\right\} \\ &\le \sup_{t\in R}\max_{(i,j)}\left\{\frac{L_{ij}^+}{a_{ij}^-}\right\} = \max_{(i,j)}\left\{\frac{L_{ij}^+}{a_{ij}^-}\right\} = L.\end{aligned} \tag{3.4}$$

Therefore, for any $\varphi \in B^*$, we have

$$\|\varphi\| \le \|\varphi - \varphi_0\| + \|\varphi_0\| \le \frac{\gamma}{1-\gamma}L + L = \frac{L}{1-\gamma}. \tag{3.5}$$

Now, we will show that $\mathcal{T}$ maps B^* into itself. In fact, for any $\varphi \in B^*$, by using $L/(1-\gamma) \le 1$, we have

$$\begin{aligned}
\left\|\mathcal{T}\varphi - \varphi_0\right\|_\infty &= \sup_{t\in R}\max_{(i,j)}\left\{\left|\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta}\left[-\sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(s) f_{ij}\left(\varphi_{hl}(s)\right)\varphi_{ij}(s)\right.\right.\right. \\
&\qquad \left.\left.\left. -\sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(s) g_{ij}\left(\varphi_{hl}(s-\tau_{hl}(s))\right)\varphi_{ij}(s)\right]ds\right|\right\} \\
&\le \sup_{t\in R}\max_{(i,j)}\left\{\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(s)\left|\left(\varphi_{hl}(s)\right)\right|\ \left|\varphi_{ij}(s)\right|\right.\right. \\
&\qquad \left.\left. +\nu_{ij}\sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(s)\left|\varphi_{hl}(s-\tau_{hl}(s))\right|\left|\varphi_{ij}(s)\right|\right]ds\right\} \\
&\le \max_{(i,j)}\left\{\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta}\left(\mu_{ij}\sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(s) + \nu_{ij}\sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(s)\right)ds\right\}\|\varphi\|^2. \\
&\le \gamma\|\varphi\| \le \gamma\|\varphi\| \le \frac{\gamma L}{1-\gamma}.
\end{aligned} \tag{3.6}$$

For any $\varphi, \phi \in B^*$, it follows from $L/(1-\gamma) \le 1$ that

$$\begin{aligned}
&\left\|\mathcal{T}\varphi - \mathcal{T}\psi\right\|_\infty \\
&= \sup_{t\in R}\max_{(i,j)}\left\{\left|\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta}\right.\right. \\
&\qquad \times\left[\sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(s)\left|f_{ij}\left(\varphi_{hl}(s)\right)\varphi_{ij}(s) - f_{ij}\left(\psi_{hl}(s)\right)\psi_{ij}(s)\right|\right. \\
&\qquad \left.\left.\left. +\sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(s)\left|g_{ij}\left(\varphi_{hl}(s-\tau_{hl}(s))\right)\varphi_{ij}(s) - g_{ij}\left(\psi_{hl}(s-\tau_{hl}(s))\right)\psi_{ij}(s)\right|\right]ds\right|\right\} \\
&\le \sup_{t\in R}\max_{(i,j)}\left\{\int_{-\infty}^{t} e^{-\int_s^t a_{ij}(\eta)d\eta}\right. \\
&\qquad \left.\times\left[\sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(s)\left(\left|f_{ij}\left(\varphi_{hl}(s)\right)\right|\cdot\left|\varphi_{ij}(s) - \psi_{ij}(s)\right|\right.\right.\right.
\end{aligned}$$

$$+\left|f_{ij}(\varphi_{hl}(s))-f_{ij}(\psi_{hl}(s))\right|\cdot\left|\varphi_{ij}(s)\right|)+\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(s)\left(\left|g_{ij}(\varphi_{hl}(s-\tau_{hl}(s)))\right|\right.$$

$$\left.\times\left|\varphi_{ij}(s)-\psi_{ij}(s)\right|+\left|g_{ij}(\varphi_{hl}(s-\tau_{hl}(s)))-g_{ij}(\psi_{hl}(s-\tau_{hl}(s)))\right|\cdot\left|\varphi_{ij}(s)\right|\right)]ds\Bigg\}$$

$$\leq\max_{(i,j)}\left\{\int_{-\infty}^{t}e^{-\int_s^t a_{ij}(\eta)d\eta}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(s)+\nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(s)\right](\|\varphi\|_\infty+\|\psi\|_\infty)\|\varphi-\psi\|ds\right\}$$

$$=\gamma\cdot\frac{2L}{1-\gamma}\|\varphi-\psi\|=\frac{2L\gamma}{1-\gamma}\|\varphi-\psi\|=\delta\|\varphi-\psi\|. \tag{3.7}$$

Since $\delta<1$, $\mathcal{T}$ is a contraction mapping. Therefore, there exists a unique fixed point $x^*\in B^*$ such that $\mathcal{T}x^*=x^*$. That is, system (1.4) has a unique pseudo almost-periodic solution $x^*\in B^*$ with $\|x^*-\varphi_0\|\leq(\gamma/(1-\gamma))L$. □

Now we go ahead with the GES of (1.4). The approaches involve constructing suitable Lyapunov functions and application of a generalized Halanay's delay differential inequality. We will stop here to see our first criteria for the globally exponential stability of (1.4), which is delay dependent.

Theorem 3.2. *In addition to (A_1)-(A_4), if one further assumes that*

(A_5)

$$c=\min_{(i,j)}\inf_{(i,j)}\left\{\beta_{ij}\left(2a_{ij}(t)-3\beta_{ij}\frac{L}{1-\gamma}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)+\nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\right]\right)\right.$$

$$\left.+\beta_{hl}\left(\mu_{hl}\frac{L}{1-\gamma}\sum_{B^{ij}\in N_r(h,l)}B_{hl}^{ij}(t)+\frac{L}{1-\gamma}\nu_{hl}\sum_{C^{hl}\in N_r(i,j)}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\right)\right\}>0 \tag{3.8}$$

or

(A_6)

$$c=\inf_{(i,j)}\left\{\beta_{ij}\left(a_{ij}(t)-\beta_{ij}\frac{L}{1-\gamma}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)+\nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\right]\right)\right.$$

$$\left.-\beta_{hl}\left(\mu_{hl}\frac{L}{1-\gamma}\sum_{B^{ij}\in N_r(h,l)}B_{hl}^{ij}(t)+\frac{L}{1-\gamma}\nu_{hl}\sum_{C^{hl}\in N_r(i,j)}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\right)\right\}>0. \tag{3.9}$$

Then there exists a unique pseudo-almost periodic solution of system (1.4) *and all other solutions converge exponentially to the (pseudo) almost-periodic attractor.*

Proof. By Theorem (3.2), there exists a unique pseudo almost-periodic solution, namely, $x(t) = x(t,\varphi)$. Let $y = y(t,\varphi)$ be any other solution of (1.4) through (t_0,φ). Assume that (A_5) is satisfied and consider the auxiliary functions $F_{ij}(\varepsilon)$ defined on $[0,+\infty]$ as follows:

$$F_{ij}(\varepsilon) = \inf_{t\in R}\left\{\beta_{ij}(2a_{ij}(t)-\varepsilon) - 3\beta_{ij}\frac{L}{1-\gamma}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t) + \nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\right]\right.$$
$$\left. +\beta_{hl}\left(\mu_{hl}\frac{L}{1-\gamma}\sum_{B^{ij}\in N_r(h,l)}B_{hl}^{ij}(t) + \frac{L}{1-\gamma}\nu_{hl}\sum_{C^{hl}\in N_r(i,j)}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\right)\right\}. \tag{3.10}$$

From (A_1)–(A_3), one can easily show that $F_{ij}(\varepsilon)$ is well defined and is continuous. From (A_5), it follows that $F_{ij}(0) > 0, F_{ij}(\varepsilon) \to -\infty$ as $\varepsilon \to \infty$ it follows that there exists an $\varepsilon_{ij} > 0$ such that $F_{ij}(\varepsilon_{ij}) > 0$. Let $\varepsilon = \min_{i,j}\varepsilon_{ij}$. Then we have $F_{ij}(\varepsilon) > 0, 1 \le i \le n, 1 \le j \le m$.

Consider the Lyapunov functional defined by

$$V(t) = \frac{1}{2}\sum_{(i,j)}\beta_{ij}\left[\left(x_{ij}(t)-y_{ij}(t)\right)^2 e^{\varepsilon t}\right.$$
$$\left. + \sum_{C^{hl}\in N_r(i,j)}V_{ij}\frac{L}{1-\gamma}\int_{t-\tau_{hl}(t)}^{t}\frac{C^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}\right)(t)}\left(x_{hl}(s)-y_{hl}(s)\right)^2 e^{\varepsilon(s+\tau_{hl}^M)}ds\right]. \tag{3.11}$$

Calculating the upper-right derivative of $V(t)$ and using the inequality $2ab \le a^2 + b^2$, one has

$$V'(t) \le \sum_{(i,j)}\beta_{ij}\left\{\frac{1}{2}\left(x_{ij}(t)-y_{ij}(t)\right)^2\varepsilon e^{\varepsilon t} - e^{\varepsilon t}a_{ij}(t)\left(x_{ij}-y_{ij}(t)\right)^2\right.$$
$$+ e^{\varepsilon t}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)\left|x_{ij}(t)-y_{ij}(t)\right|\cdot\left|f_{ij}(x_{hl}(t))x_{ij}(t) - f_{ij}\left(y_{hl}(t)\right)y_{ij}(t)\right|$$
$$+ e^{\varepsilon t}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\left|x_{ij}(t)-y_{ij}(t)\right|$$
$$\cdot\left|g_{ij}(x_{hl}(t-\tau_{hl}(t)))x_{ij}(t) - g_{ij}\left(y_{hl}(t-\tau_{hl}(t))\right)y_{ij}(t)\right|$$
$$+ \frac{1}{2}\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}\frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)}\left(x_{hl}(t)-y_{hl}(t)\right)^2 e^{\varepsilon(t+\tau_{hl}^M)}$$
$$\left. -\frac{1}{2}\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}C_{ij}^{hl}(t)\left(x_{hl}(t-\tau_{hl}(t))-y_{hl}(t-\tau_{hl}(t))\right)^2 e^{\varepsilon(t-\tau_{hl}(t)+\tau_{hl}^M)}\right\}$$

$$
\begin{aligned}
&\le \sum_{(i,j)} \beta_{ij} \Bigg\{ \frac{1}{2}\left(x_{ij}(t)-y_{ij}(t)\right)^2 \varepsilon e^{\varepsilon t} - e^{\varepsilon t} a_{ij}(t)\left(x_{ij}-y_{ij}(t)\right)^2 \\
&\qquad + e^{\varepsilon t} \sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(t)\left|x_{ij}(t)-y_{ij}(t)\right| \cdot \left|f_{ij}(x_{hl}(t))x_{ij}(t) - f_{ij}\left(y_{hl}(t)\right)y_{ij}(t)\right| \\
&\qquad + e^{\varepsilon t} \sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(t)\left|x_{ij}(t)-y_{ij}(t)\right| \\
&\qquad\qquad \cdot \left|g_{ij}(x_{hl}(t-\tau_{hl}(t)))x_{ij}(t) - g_{ij}\left(y_{hl}(t-\tau_{hl}(t))\right)y_{ij}(t)\right| \\
&\qquad + \frac{1}{2} \sum_{C^{hl}\in N_r(i,j)} v_{ij} \frac{L}{1-\gamma} \frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)} \left(x_{hl}(t)-y_{hl}(t)\right)^2 e^{\varepsilon\left(t+\tau_{hl}^{M}\right)} \\
&\qquad - \frac{1}{2} \sum_{C^{hl}\in N_r(i,j)} v_{ij} \frac{L}{1-\gamma} C_{ij}^{hl}(t)\left(x_{hl}(t-\tau_{hl}(t)) - y_{hl}(t-\tau_{hl}(t))\right)^2 e^{\varepsilon(t)} \Bigg\} \\
&\le e^{\varepsilon(t)} \sum_{(i,j)} \beta_{ij} \Bigg\{ \left(\frac{\varepsilon}{2} - a_{ij}(t)\right)\left(x_{ij}(t)-y_{ij}(t)\right)^2 \\
&\qquad + \sum_{B^{hl}\in N_r(i,j)} B_{ij}^{hl}(t)\left|x_{ij}(t)-y_{ij}(t)\right| \left[\left|f_{ij}(x_{hl}(t))\right|\left|x_{ij}(t)-y_{ij}(t)\right|\right. \\
&\qquad\qquad \left. + \left|f_{ij}(x_{hl}(t)) - f_{ij}\left(y_{hl}(t)\right)\right|\left|y_{ij}(t)\right|\right] \\
&\qquad + \sum_{C^{hl}\in N_r(i,j)} C_{ij}^{hl}(t)\left|x_{ij}(t)-y_{ij}(t)\right| \\
&\qquad\quad \times \left[\left|g_{ij}(x_{hl}(t-\tau_{hl}(t)))\right|\left|x_{ij}(t)-y_{ij}(t)\right|\right. \\
&\qquad\quad \left. + \left|g_{ij}(x_{hl}(t-\tau_{hl}(t))) - g_{ij}\left(y_{hl}(t-\tau_{hl}(t))\right)\right|\left|y_{ij}(t)\right|\right] \\
&\qquad + \frac{1}{2} \sum_{C^{hl}\in N_r(i,j)} v_{ij} \frac{L}{1-\gamma} \frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)} \left(x_{hl}(t)-y_{hl}(t)\right)^2 e^{\varepsilon\tau_{hl}^{M}} \\
&\qquad - \frac{1}{2} \sum_{C^{hl}\in N_r(i,j)} v_{ij} \frac{L}{1-\gamma} C_{ij}^{hl}(t)\left(x_{hl}(t-\tau_{hl}(t)) - y_{hl}(t-\tau_{hl}(t))\right)^2 \Bigg\}
\end{aligned}
$$

$$
\begin{aligned}
&\le e^{\varepsilon(t)}\sum_{(i,j)}\beta_{ij}\Bigg\{\left(\frac{\varepsilon}{2}-a_{ij}(t)\right)\left(x_{ij}(t)-y_{ij}(t)\right)^2\\
&\qquad+\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)\mu_{ij}\frac{L}{1-\gamma}\\
&\qquad\qquad\times\left[\left(x_{ij}(t)-y_{ij}(t)\right)^2+\left|x_{ij}(t)-y_{ij}(t)\right|\left|x_{hl}(t)-y_{hl}(t)\right|\right]\\
&\qquad+\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\nu_{ij}\frac{L}{1-\gamma}\\
&\qquad\qquad\times\left[\left(x_{ij}(t)-y_{ij}(t)\right)^2+\left|x_{ij}(t)-y_{ij}(t)\right|\right.\\
&\qquad\qquad\left.\times\left|x_{hl}(t-\tau_{hl}(t))-y_{hl}(t-\tau_{hl}(t))\right|\right]\\
&\qquad+\frac{1}{2}\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}\frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)}\left(x_{ij}(t)-y_{ij}(t)\right)^2e^{\varepsilon\tau_{hl}^M}\\
&\qquad-\frac{1}{2}\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}C_{ij}^{hl}(t)\left(x_{hl}(t-\tau_{hl}(t))-y_{hl}(t-\tau_{hl}(t))\right)^2\Bigg\}\\
&\le e^{\varepsilon(t)}\sum_{(i,j)}\beta_{ij}\Bigg\{\left(\frac{\varepsilon}{2}-a_{ij}(t)\right)\left(x_{ij}(t)-y_{ij}(t)\right)^2\\
&\qquad+\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)\mu_{ij}\frac{L}{1-\gamma}\\
&\qquad\qquad\times\left[\left(x_{ij}(t)-y_{ij}(t)\right)^2+\frac{1}{2}\left(x_{ij}(t)-y_{ij}(t)\right)^2\left(x_{hl}(t)-y_{hl}(t)\right)^2\right]\\
&\qquad+\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\nu_{ij}\frac{L}{1-\gamma}\\
&\qquad\qquad\times\left[\left(x_{ij}(t)-y_{ij}(t)\right)^2+\frac{1}{2}\left(x_{ij}(t)-y_{ij}(t)\right)^2\right.\\
&\qquad\qquad\left.\times\left(x_{hl}(t-\tau_{hl}(t))-y_{hl}(t-\tau_{hl}(t))\right)^2\right]\\
&\qquad+\frac{1}{2}\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}\frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)}\left(x_{ij}(t)-y_{ij}(t)\right)^2e^{\varepsilon\tau_{hl}^M}\\
&\qquad-\frac{1}{2}\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}C_{ij}^{hl}(t)\left(x_{hl}(t-\tau_{hl}(t))-y_{hl}(t-\tau_{hl}(t))\right)^2\Bigg\}
\end{aligned}
$$

$$\begin{aligned}
&\le e^{\varepsilon(t)}\sum_{(i,j)}\left\{\beta_{ij}\left(\frac{\varepsilon}{2}-a_{ij}(t)\right)\left(x_{ij}(t)-y_{ij}(t)\right)^2+\frac{3}{2}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)\mu_{ij}\frac{L}{1-\gamma}\left(x_{ij}(t)-y_{ij}(t)\right)^2\right.\\
&\qquad+\frac{3}{2}\beta_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\nu_{ij}\frac{L}{1-\gamma}\left(x_{ij}(t)-y_{ij}(t)\right)^2\\
&\qquad+\frac{1}{2}\sum_{B^{ij}\in N_r(h,l)}\beta_{hl}B_{hl}^{ij}(t)\mu_{hl}\frac{L}{1-\gamma}\left(x_{ij}(t)-y_{ij}(t)\right)^2\\
&\qquad\left.+\frac{1}{2}\sum_{C^{ij}\in N_r(h,l)}\nu_{hl}\frac{L}{1-\gamma}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\left(x_{ij}(t)-y_{ij}(t)\right)^2e^{\varepsilon\tau_{hl}^{M}}\right\}\\
&\le-\frac{c_0}{2}e^{\varepsilon t}\left(x_{ij}(t)-y_{ij}(t)\right)^2\le 0,
\end{aligned}\tag{3.12}$$

where $c_0>0$ is defined by

$$\begin{aligned}
c_0=\min_{(i,j)}\inf_{t\in R}&\left\{\beta_{ij}\left(2a_{ij}(t)-\varepsilon\right)-3\beta_{ij}\frac{L}{1-\gamma}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)+\nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\right]\right.\\
&\left.+\beta_{hl}\left(\mu_{hl}\frac{L}{1-\gamma}\sum_{B^{ij}\in N_r(h,l)}B_{hl}^{ij}(t)+\frac{L}{1-\gamma}\nu_{hl}\sum_{C^{hl}\in N_r(i,j)}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\right)\right\}>0.
\end{aligned}\tag{3.13}$$

From the above, we have $V(t)\le V(t_0), t\ge t_0$, and

$$\begin{aligned}
&\frac{1}{2}e^{\varepsilon t}\min_{h,l}\beta_{hl}\sum_{i=1}^{n}\left(x_{ij}(t)-y_{ij}(t)\right)^2\le V(t),\quad t\ge t_0,\\
&V(t_0)\le\frac{1}{2}e^{\varepsilon t_0}\left[\sum_{(i,j)}\beta_{ij}+\sum_{(i,j)}\sum_{(h,l)}\beta_{ij}\nu_{ij}\frac{L}{1-\gamma}\tau_{hl}^{\mu}e^{\varepsilon\tau_{hl}^{\mu}}\sup_{t\in[t_0-\tau,t_0]}\frac{C^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)}\right]\left\|\varphi-\varphi\right\|_2^2.
\end{aligned}\tag{3.14}$$

Thus, it follows that there exists a positive constant $M>1$ such that

$$\left|x(t)-y(t)\right|_2\le Me^{-(\varepsilon/2)(t-t_0)}\left\|\varphi-\varphi\right\|_2,\quad t\ge t_0,\tag{3.15}$$

which implies that (1.4) is GES.

Now we assume that (A_6) is satisfied. By carrying out similar arguments as above, one can easily show that there exists an $\varepsilon > 0$ such that

$$\inf_{(i,j)}\left\{\beta_{ij}\left(a_{ij}(t)-\varepsilon\right)-\beta_{ij}\frac{L}{1-\gamma}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)+\nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\right]\right.$$
$$\left.-\beta_{hl}\left(\mu_{hl}\frac{L}{1-\gamma}\sum_{B^{ij}\in N_r(h,l)}B_{hl}^{ij}(t)+\frac{L}{1-\gamma}\nu_{hl}\sum_{C^{hl}\in N_r(i,j)}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\right)\right\}>0. \tag{3.16}$$

Consider the Lyapunov function

$$V(t)=\sum_{(i,j)}\beta_{ij}\left\{\left|x_{ij}(t)-y_{ij}(t)\right|e^{\varepsilon t}+\sum_{C^{hl}\in N_r(i,j)}\nu_{ij}\frac{L}{1-\gamma}\right.$$
$$\left.\times\int_{t-\tau_{hl}(t)}^{t}\frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(s)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(s)\right)}\left|x_{hl}(s)-y_{hl}(s)\right|e^{\varepsilon\left(t+\tau_{hl}^{\mu}\right)}ds\right\}. \tag{3.17}$$

Similar to the above arguments, calculating the upper-right derivative $D^+V(t)$ produces

$$D^+V(t)\le -c_2e^{\varepsilon t}\sum_{(i,j)}\left|x_{ij}(t)-y_{ij}(t)\right|\le 0, \tag{3.18}$$

where

$$c_2=\min_{(i,j)}\inf_{t\in R}\left\{\beta_{ij}\left(a_{ij}(t)-\varepsilon\right)-\beta_{ij}\frac{L}{1-\gamma}\left[\mu_{ij}\sum_{B^{hl}\in N_r(i,j)}B_{ij}^{hl}(t)+\nu_{ij}\sum_{C^{hl}\in N_r(i,j)}C_{ij}^{hl}(t)\right]\right.$$
$$\left.-\beta_{hl}\left(\mu_{hl}\frac{L}{1-\gamma}\sum_{B^{ij}\in N_r(h,l)}B_{hl}^{ij}(t)+\frac{L}{1-\gamma}\nu_{hl}\sum_{C^{hl}\in N_r(i,j)}\frac{C_{hl}^{ij}\left(\xi_{ij}^{-1}(t)\right)}{1-\tau_{ij}\left(\xi_{ij}^{-1}(t)\right)}\right)\right\}>0. \tag{3.19}$$

Then we have

$$e^{\varepsilon t}\left(\min_{(h,l)}\beta_{hl}\right)\sum_{(i,j)}\left|x_{ij}(t)-y_{ij}(t)\right|\le V(t)\le V(t_0),\quad t\ge t_0. \tag{3.20}$$

Note that

$$V(t_0)\le e^{\varepsilon t_0}\left[\sum_{(i,j)}\beta_{ij}+\sum_{(i,j)}\beta_{ij}\sum_{C^{ij}\in N_r(h,l)}\tau_{hl}^{\mu}e^{\varepsilon\tau_{hl}^{\mu}}\sup_{t\in[t_0-\tau,t_0]}\frac{C_{ij}^{hl}\left(\xi_{hl}^{-1}(t)\right)}{1-\tau_{hl}\left(\xi_{hl}^{-1}(t)\right)}\right]\left\|\varphi-\psi\right\|_1. \tag{3.21}$$

Then there exists a positive constant $M > 1$ such that

$$\left|x(t) - y(t)\right|_1 = \sum_{i=1}^{n} \left|x_i(t) - y_i(t)\right| \leq M \left\|\varphi - \psi\right\|_1 e^{\varepsilon(t-t_0)}, \quad t \geq t_0. \tag{3.22}$$

The proof is complete. □

References

[1] A. Bouzerdoum and R. B. Pinter, "Shunting inhibitory cellular neural networks: derivation and stability analysis," *IEEE Transactions on Circuits and Systems I*, vol. 40, no. 3, pp. 215–221, 1993.

[2] Y. Liu, Z. You, and L. Cao, "On the almost periodic solution of generalized shunting inhibitory cellular neural networks with continuously distributed delays," *Physics Letters A*, vol. 360, no. 1, pp. 122–130, 2006.

[3] T. Zhou, Y. Liu, and A. Chen, "Almost periodic solution for shunting inhibitory cellular neural networks with time-varying delays and variable coefficients," *Neural Processing Letters*, vol. 23, no. 3, pp. 243–255, 2006.

[4] L. Chen and H. Zhao, "Global stability of almost periodic solution of shunting inhibitory cellular neural networks with variable coefficients," *Chaos, Solitons & Fractals*, vol. 35, no. 2, pp. 351–357, 2008.

[5] A. Chen and J. Cao, "Almost periodic solution of shunting inhibitory CNNs with delays," *Physics Letters A*, vol. 298, no. 2-3, pp. 161–170, 2002.

[6] Y. Li, C. Liu, and L. Zhu, "Global exponential stability of periodic solution for shunting inhibitory CNNs with delays," *Physics Letters A*, vol. 337, no. 1-2, pp. 46–54, 2005.

[7] X. Huang and J. Cao, "Almost periodic solution of shunting inhibitory cellular neural networks with time-varying delay," *Physics Letters A*, vol. 314, no. 3, pp. 222–231, 2003.

[8] B. Liu and L. Huang, "Existence and stability of almost periodic solutions for shunting inhibitory cellular neural networks with time-varying delays," *Chaos, Solitons & Fractals*, vol. 31, no. 1, pp. 211–217, 2007.

[9] A. Chen and J. Cao, "Almost periodic solution of shunting inhibitory CNNs with delays," *Physics Letters A*, vol. 298, no. 2-3, pp. 161–170, 2002.

[10] Y. Xia, J. Cao, and Z. Huang, "Existence and exponential stability of almost periodic solution for shunting inhibitory cellular neural networks with impulses," *Chaos, Solitons & Fractals*, vol. 34, no. 5, pp. 1599–1607, 2007.

[11] Q. Zhou, B. Xiao, Y. Yu, and L. Peng, "Existence and exponential stability of almost periodic solutions for shunting inhibitory cellular neural networks with continuously distributed delays," *Chaos, Solitons & Fractals*, vol. 34, no. 3, pp. 860–866, 2007.

[12] B. Liu, "Almost periodic solutions for shunting inhibitory cellular neural networks without global Lipschitz activation functions," *Journal of Computational and Applied Mathematics*, vol. 203, no. 1, pp. 159–168, 2007.

[13] C. Y. Zhang, *Pseudo almost periodic functions and their applications*, Ph.D. thesis, University of Western Ontario, Ontario, Canada, 1992.

[14] C. Y. Zhang, "Pseudo-almost-periodic solutions of some differential equations," *Journal of Mathematical Analysis and Applications*, vol. 181, no. 1, pp. 62–76, 1994.

[15] C. Y. He, *Almost Periodic Differential Equations*, Higher Education Publishing House, Beijing, China, 1992.

[16] A. M. Fink, *Almost Periodic Differential Equations*, vol. 377 of *Lecture Notes in Mathematics*, Springer, Berlin, Germany, 1974.

Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2011, Article ID 301204, 34 pages
doi:10.1155/2011/301204

Research Article

Encoding Static and Temporal Patterns with a Bidirectional Heteroassociative Memory

Sylvain Chartier[1] and Mounir Boukadoum[2]

[1] *School of Psychology, University of Ottawa, 136 Jean-Jacques Lussier, Ottawa, ON, Canada K1N 6N5*
[2] *Département d'informatique, Université du Québec à Montréal, Case postale 8888, Succursale Centre-ville, Montréal, QC, Canada H3C 3P8*

Correspondence should be addressed to Sylvain Chartier, schartie@uottawa.ca

Received 15 September 2010; Accepted 8 December 2010

Academic Editor: Haydar Akca

Brain-inspired, artificial neural network approach offers the ability to develop attractors for each pattern if feedback connections are allowed. It also exhibits great stability and adaptability with regards to noise and pattern degradation and can perform generalization tasks. In particular, the Bidirectional Associative Memory (BAM) model has shown great promise for pattern recognition for its capacity to be trained using a supervised or unsupervised scheme. This paper describes such a BAM, one that can encode patterns of real and binary values, perform multistep pattern recognition of variable-size time series and accomplish many-to-one associations. Moreover, it will be shown that the BAM can be generalized to multiple associative memories, and that it can be used to store associations from multiple sources as well. The various behaviors are the result of only topological rearrangements, and the same learning and transmission functions are kept constant throughout the models. Therefore, a consistent architecture is used for different tasks, thereby increasing its practical appeal and modeling importance. Simulations show the BAM's various capacities, by using several types of encoding and recall situations.

1. Introduction

Being able to recognize and recall patterns of various natures and in different contexts is something that human beings accomplish routinely and with little effort. But these tasks are difficult to reproduce by artificial intelligent systems. A successful approach has consisted of distributing information over parallel networks of processing units, as done in biological neural networks. This brain-inspired, artificial neural network approach offers the ability to develop attractors for each pattern if feedback connections are allowed (e.g., [1, 2]). It also exhibits great stability and adaptability with regard to noise and pattern degradation, and can perform generalization tasks. In particular, the Bidirectional Associative

Memory (BAM) model has shown great promise for pattern recognition for its capacity to be trained using a supervised or unsupervised scheme [3]. Given n bipolar colon-matrix pairs, $(\mathbf{X}_1, \mathbf{Y}_1), (\mathbf{X}_2, \mathbf{Y}_2), \ldots, (\mathbf{X}_i, \mathbf{Y}_i), \ldots, (\mathbf{X}_n, \mathbf{Y}_n)$, BAM learning is accomplished with a simple Hebbian rule, according to the equation:

$$\mathbf{W} = \left(\mathbf{Y}_1\mathbf{X}_1^T\right) + \left(\mathbf{Y}_2\mathbf{X}_2^T\right) + \cdots + \left(\mathbf{Y}_n\mathbf{X}_n^T\right) = \mathbf{Y}\mathbf{X}^T. \tag{1.1}$$

In this expression, $\mathbf{X}$ and $\mathbf{Y}$ are matrices that represent the sets of bipolar pairs to be associated, and $\mathbf{W}$ is the weight matrix. To assure perfect storage and retrieval, the input patterns needed to be orthogonal ($\mathbf{X}^T\mathbf{X} = \mathbf{K}$, with typically $\mathbf{K} = \mathbf{I}$ the identity matrix; [4]). In that case, all the positive eigenvalues of the weight matrix will be equal and, when an input is presented for recall, the correct output can be retrieved. For example, if the input $\mathbf{x}$ represents the encoded pattern $\mathbf{X}_i$, then the output will be given by

$$\begin{aligned}\mathbf{y} &= \mathbf{W}\mathbf{x} = \mathbf{Y}\mathbf{X}^T\mathbf{x} \\ &= \mathbf{Y}_1\mathbf{X}_1^T\mathbf{x} + \mathbf{Y}_2\mathbf{X}_2^T\mathbf{x} + \cdots + \mathbf{Y}_i\mathbf{X}_i^T\mathbf{x} + \cdots + \mathbf{Y}_n\mathbf{X}_n^T\mathbf{x} \\ &= \mathbf{Y}_1 0 + \mathbf{Y}_2 0 + \cdots + \mathbf{Y}_i 1 \cdots + \mathbf{Y}_n 0 \\ &= \mathbf{Y}_i.\end{aligned} \tag{1.2}$$

The $\mathbf{y}$ output will thus correspond to the $\mathbf{Y}_i$ encoded stimulus. Equation (1.1) uses a one-shot learning process since Hebbian association is strictly additive. A more natural learning procedure would make the learning incremental but, then, the weight matrix would grow unbounded with the repetition of the input stimuli during learning. In addition, the eigenvalues will reflect the frequency of presentation of each stimulus. This property may be acceptable for orthogonal patterns, but it leads to disastrous results when the patterns are correlated. In that case, the weight matrix will be dominated by its first eigenvalue, and this will result in recalling the same pattern whatever the input. For correlated patterns, a compromise is to use a one-shot learning rule to limit the domination of the first eigenvalue, and to use a recurrent nonlinear transmission function to allow the network to filter out the different patterns during recall. Kosko's BAM effectively used a signum transmission function to recall noisy patterns, despite the fact that the weight matrix developed by using (1.1) is not optimal. The nonlinear transmission function usually used by the BAM network is expressed by the following equations:

$$\begin{aligned}\mathbf{y}(t+1) &= f(\mathbf{W}\mathbf{x}(t)), \\ \mathbf{x}(t+1) &= f\left(\mathbf{W}^T\mathbf{x}(t)\right),\end{aligned} \tag{1.3}$$

where $f(\,)$ is the signum function and t is a given discrete time step. Initially, BAMs had poor storage capacity, could only store binary patterns, and were limited to static inputs. Nowadays, storage and recall performance have much improved; BAMs can learn and recall binary patterns with good performance (e.g., [5–20]), and that capability has been extended to real-valued patterns (e.g., [7, 8, 21–24]). Attention has also been given to learning and retrieving temporal sequences (see [25] for a review). These multistep associations have been

studied essentially with gradient descent techniques (e.g., [26, 27]), competitive techniques (e.g., [28]) or complex architectures (e.g., [29, 30]). Few authors have studied this type of association in the case of simple BAM networks (e.g., [7, 31–33]).

In this paper, focus will be drawn on BAM properties for different kinds of associations. At first analysis of the transmission function will be studied both analytically and numerically. It will be followed by simulations of the classic case of one-to-one association. The next section will present many-to-one association that allows mean extraction from the presentation of different exemplars. Then, we focus on temporal association. In this case, both limit cycle and steady-state behaviors will be presented. Finally, more complex architectures will be explored by using the Multidirectional Associative Memory (MAM) architecture while keeping the same learning and transmission functions.

2. Analysis of the Transmission Function

This section describes how the transmission function is derived from both analytical and numerical results of the one-dimensional cubic function.

2.1. One-Dimensional Symmetric Setting

The transmission function used in the model is based on the classic Verhulst equation [34]. Since, this quadratic function has only one stable fixed-point, it is extended to a cubic form described by the dynamic equation

$$\frac{dx}{dt} = f(x) = rx - x^3, \tag{2.1}$$

where x represents the input value, and r is a free parameter that affects the equilibrium states of (2.1). Figure 1 illustrates its shape for $r = 1$. The fixed-points are the roots of $f(x) = 0$. For example, if $r = 1$ the corresponding fixed-points are $\overline{x} = -1$, 0, and 1. The stability properties of the fixed-points are determined by computing the derivative of (2.1):

$$f'(x) = r - 3x^2. \tag{2.2}$$

If the derivative at a given fixed-point is greater than zero, then a small perturbation results in growth; else, if the derivative is negative, then a small perturbation results in decay. The first situation represents an unstable fixed-point whereas the second represents a stable one. In the previous example, both $\overline{x} = 1$ and $\overline{x} = -1$ are stable fixed-points while $\overline{x} = 0$ is an unstable fixed-point. This is illustrated in Figure 1 by filled and empty circles, respectively.

Another way to visualize the dynamics of a recurrent neural network is based on the physical idea of energy [1]. The energy function, noted $E(x)$, can be defined by

$$-\frac{dE}{dx} = \frac{dx}{dt}, \tag{2.3}$$

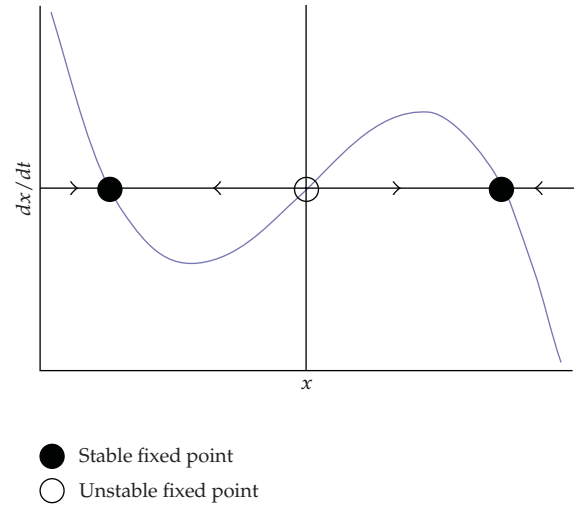


Figure 1: Transmission function when $r = 1$ showing two stable and one unstable fixed-points.

where the negative sign indicates that the state vector moves downhill in the energy landscape. Using the chain rule, it follows from (2.3) that

$$\frac{dE}{dt} = \frac{dE}{dx}\left(-\frac{dE}{dx}\right) = -\left(\frac{dE}{dx}\right)^2 \leq 0. \tag{2.4}$$

Thus, $E(t)$ decreases along trajectories or, in other words, the state vector globally converges towards lower energy states. Equilibrium occurs at locations of the vector field where local minima correspond to stable fixed-points, and local maxima correspond to unstable ones. To find them, we need to find $E(x)$ such that

$$-\frac{dE}{dx} = rx - x^3. \tag{2.5}$$

The general solution is

$$E(x) = -\frac{rx^2}{2} + \frac{x^4}{4} + C, \tag{2.6}$$

where C is an arbitrary constant (we use $C = 0$ for convenience). Figure 2 illustrates the energy function when $r = 1$. The system exhibits a double-well potential with two stable equilibrium points ($\overline{x} = -1$ and $\overline{x} = 1$).

The r parameter plays an important role in determining the number and positions of the fixed-points. Figure 3 illustrates the situation. For r less than zero, the system has only one (stable) fixed-point, $\overline{x} = 0$; for r greater than zero, there are three fixed-points: $\overline{x} = 0$ and $\overline{x} = \pm\sqrt{r}$, of which the first one is unstable and the other two are stable. Finally, for r equal to zero, we have a pitchfork bifurcation point. We deduce from the preceding that we must have $r > 0$ for the system to store binary stimuli.

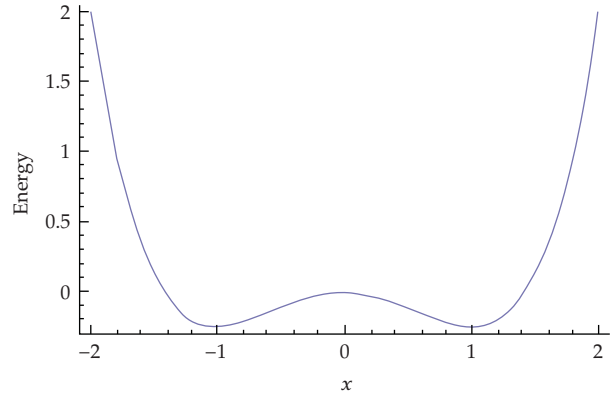


Figure 2: Energy landscape of a 1-dimensional system with $r = 1$.

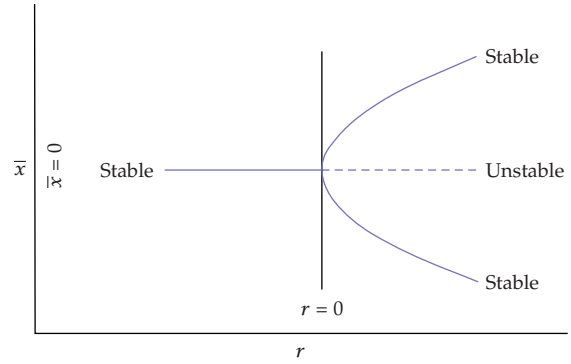


Figure 3: Phase diagram indicating the presence of a bifurcation for $r = 0$.

2.2. M-Dimensional Symmetric Setting

In a m-dimensional space, (2.1) takes a vector form and becomes

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}) = \mathbf{r} * \mathbf{x} - \mathbf{x}^3, \tag{2.7}$$

where, $\mathbf{x} = (x_1, x_2, \ldots, x_m)^T$, and $\mathbf{r} = (r_1, r_2, \ldots, r_m)^T$ and m is the network's dimension. When the weight matrix is diagonal, the system is uncoupled and the analysis is straightforward. As in the one-dimensional system, the fixed-points are defined by the roots of (2.7). Their stability properties of each one are determined by finding the eigenvalues of its Jacobian matrix. Depending on the eigenvalues, different types of fixed-point behaviors are obtained. For example, if $\mathbf{r} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ there will be nine fixed-points as illustrated in Figure 4. Four of them are stable: $(-1, -1)$, $(-1, 1)$, $(1, -1)$, and $(1, 1)$; they are indicated by filled circles. The other five are unstable and indicated by empty circles. Here also, the stability of the fixed-point can

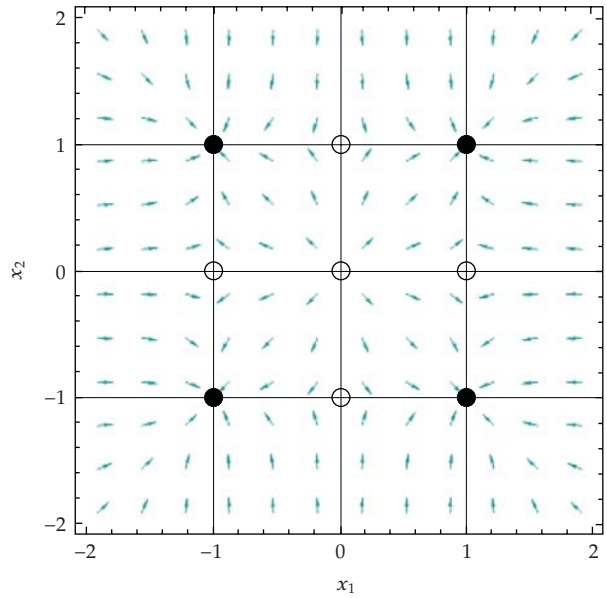


Figure 4: Phase portrait of a two-dimensional system with $\mathbf{r} = [1,1]^T$. The stable fixed-points are represented by filled circles and the unstable ones by empty circles.

be determined from the energy of the system. In matrix notation, the energy function is

$$E(x) = -r * \frac{\mathbf{x}^T\mathbf{x}}{2} + \frac{\mathbf{x}^T\mathbf{x}^3}{4}. \tag{2.8}$$

In the case of bipolar pattern and if $\mathbf{r}$ is set to $[1,1]^T$, then the energy of the system will be equal to $-m/4$.

The function is plotted in Figure 5 for a two-dimensional system, using the previous values of $\mathbf{r}$. The stable fixed-points correspond to the local minima in the plot and they partition the recall space is into four equal wells. For example, if the desired correct output (fixed-point) is $\overline{\mathbf{x}} = [1,1]^T$, the probability that this fixed-point attracts a uniformly distributed pattern $\mathbf{x}$ whose elements $x_i \in [-1,1]$ is 25%.

2.3. *Numerical Approximation Using Euler's Method*

From the previous analysis, two stable fixed-points exist when $\mathbf{r}_i$ is positive. For simplicity, if all the element of $\mathbf{r}$ are set to one, then (2.7) becomes

$$\frac{d\mathbf{x}}{dt} = \mathbf{x} - \mathbf{x}^3. \tag{2.9}$$

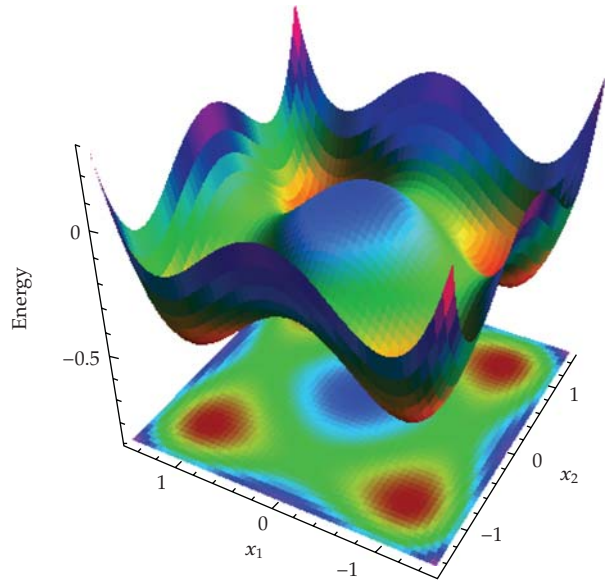


Figure 5: Energy landscape of the same two-dimensional system as in Figure 4.

This last differential equation can then be approximated in discrete time by using the Euler method:

$$\mathbf{x}(t+1) = \delta\left(\mathbf{x}(t) - \mathbf{x}^3(t)\right) + \mathbf{x}(t), \tag{2.10}$$

where δ is small positive constant and t is a given discrete time step. Rearranging the terms yields

$$\mathbf{x}(t+1) = (\delta+1)\mathbf{x}(t) - \delta\mathbf{x}^3(t). \tag{2.11}$$

However, as it is, (2.11) does not reflect the impact of weight connections. To take into account that the connection strength between units is modified by a learning rule, we pose $\mathbf{x}(t) = \mathbf{W}\mathbf{x}(t) = \mathbf{a}(t)$, where $\mathbf{W}$ represents the weight connections. Equation (2.11) then becomes

$$\mathbf{x}(t+1) = (\delta+1)\mathbf{a}(t) - \delta\mathbf{a}^3(t). \tag{2.12}$$

This last function is illustrated in Figure 6. As the figure shows, $\mathbf{a}_i(t) = 1$, then $\mathbf{x}_i(t+1)$ will have the same value of 1.

The slope of the derivative of the transmission function will determine its stability. In our case, it is desired that the fixed-points have a stable monotonic approach. Therefore, the

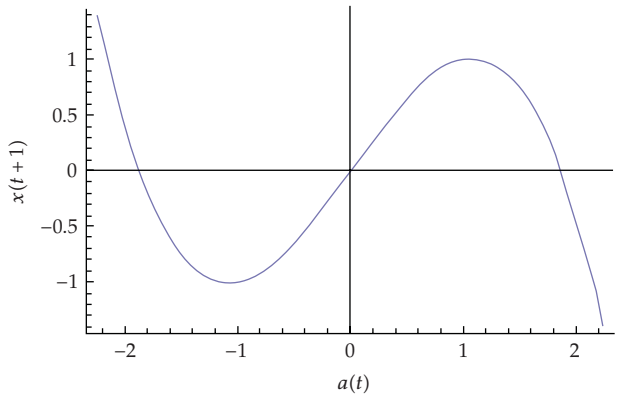


Figure 6: Transmission function for a value of $\delta = 0.4$.

slope must be positive and less than one [35]:

$$0 < \frac{d\mathbf{x}(t+1)}{d\mathbf{a}(t)} < 1 = 0 < (\delta + 1) - 3\delta \mathbf{a}^2(t) < 1. \tag{2.13}$$

This condition is satisfied when $0 < \delta < 0.5$ for bipolar stimuli. In that case, $\mathbf{a} = \mathbf{W}\,\overline{\mathbf{x}}(t) = \pm 1$. Another way to analyze the behavior of the network for the whole range of δ is by performing a Lyapunov Analysis. This analysis will allow us to discriminate between aperiodic (chaos) attractor from periodic one. The Lyapunov exponent for the case of a one-dimensional network is approximated by

$$\lambda \approx \frac{1}{T} \sum_{t=1}^{T} \log \left| \frac{d\mathbf{x}(t+1)}{d\mathbf{x}(t)} \right|, \tag{2.14}$$

where T is the number of network iterations, set to 10,000 to establish the approximation. Again, for simplicity, we perform the analysis in the case of independent units. In other words, the derivative term is obtained from (2.13) when $\mathbf{W} = \mathbf{I}$ (for simplicity), so that λ is given by

$$\lambda \approx \frac{1}{T} \sum_{t=1}^{T} \log \left| 1 + \delta - 3\delta \mathbf{x}^2(t) \right|. \tag{2.15}$$

In order to estimate the range of values for a given period, a bifurcation diagram can be performed. Figure 7 illustrates both Lyapunov and bifurcation analysis and shows that for values of δ less than 1.0, the network converges to a fixed-point attractor. However, at higher values (e.g., 1.6), the network may converge to an aperiodic attractor.

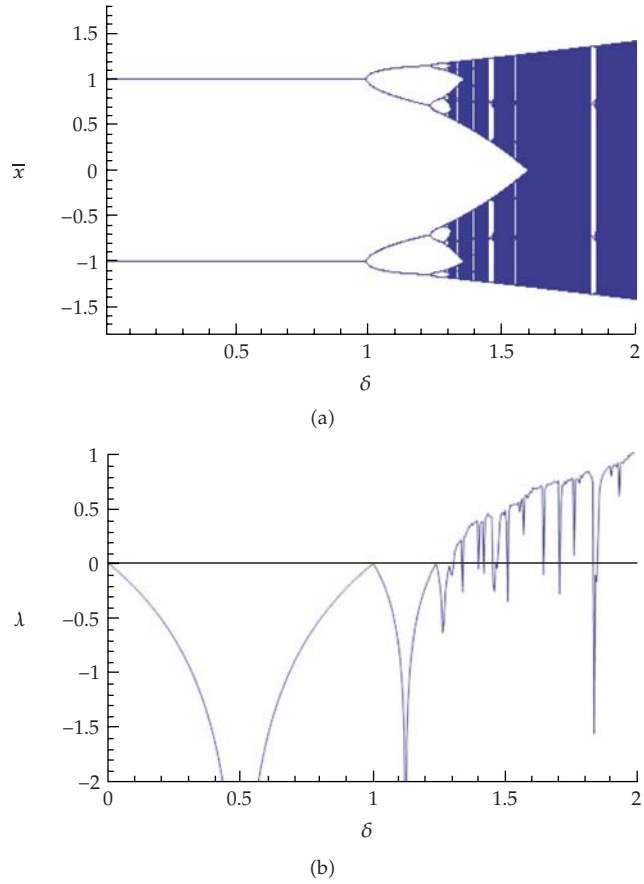


Figure 7: Bifurcation and Lyaponov exponent diagrams as a function of δ.

2.4. Numerical Approximation Using Forth-Order Runge Kutta's Method

Another numerical approximation of the network's dynamics is by using the Forth-Order Runge Kutta (FORK) method. Contrarily to Euler's method, FORK uses an average estimation to approximate the next time step

$$\begin{aligned}
\mathbf{x}(t+1) &= \mathbf{x}(t) + \frac{k1 + 2k2 + 2k3 + k4}{6}, \\
k1 &= \left(r\mathbf{x}(t) - \mathbf{x}(t)^3\right)\Delta, \\
k2 &= \left(r(\mathbf{x}(t) + 0.5k1) - (\mathbf{x}(t) + 0.5k1)^3\right)\Delta, \\
k3 &= \left(r(\mathbf{x}(t) + 0.5k2) - (\mathbf{x}(t) + 0.5k2)^3\right)\Delta\ , \\
k4 &= \left(r(\mathbf{x}(t) + k3) - (\mathbf{x}(t) + k3)^3\right)\Delta\ ,
\end{aligned} \tag{2.16}$$

where Δ is a small approximation parameter. Again, to take into account weight connections, we pose that $\mathbf{x}(t) = \mathbf{W}\mathbf{x}(t) = \mathbf{a}(t)$. Equation (2.16) thus becomes

$$\begin{aligned}
\mathbf{x}(t+1) &= \mathbf{a}(t) + \frac{k1 + 2k2 + 2k3 + k4}{6}, \\
k1 &= \left(r\mathbf{a}(t) - \mathbf{a}(t)^3\right)\Delta, \\
k2 &= \left(r(\mathbf{a}(t) + 0.5k1) - (\mathbf{a}(t) + 0.5k1)^3\right)\Delta, \\
k3 &= \left(r(\mathbf{a}(t) + 0.5k2) - (\mathbf{a}(t) + 0.5k2)^3\right)\Delta, \\
k4 &= \left(r(\mathbf{a}(t) + k3) - (\mathbf{a}(t) + k3)^3\right)\Delta.
\end{aligned} \tag{2.17}$$

This last function is illustrated in Figure 8. As the figure shows, if $\mathbf{a}_i(t) = 1$, then $\mathbf{x}_i(t+1)$ will have the same value of 1. Again, like any nonlinear dynamic system, to guarantee that a given output converges to a fixed-point $\overline{\mathbf{x}}(t)$, the slope of the derivative of the transmission function must be positive and less than one [35]:

$$0 < \frac{d\mathbf{x}(t+1)}{d\mathbf{a}(t)} < 1. \tag{2.18}$$

This condition is satisfied when $0 < \Delta < 0.135$ for bipolar stimuli. In that case, $\mathbf{a} = \mathbf{W}\overline{\mathbf{x}}(t) = \pm 1$. Here also, bifurcation and Lyapunov exponent diagrams were performed. Figure 8 shows that if the value of Δ is lower than 1.7, the network will converge to a fixed-point. However, if the value is too high then a given input will converge to an aperiodic attractor.

2.5. Performance Comparison between Euler and FORK Approximations

The purpose of this simulation was to compare the performance between Euler and FORK approximations. Although FORK gives a more precise approximation of ordinary differential equation (2.7), we need to evaluate if a better approximation translates into a better radius of attraction.

2.5.1. Methodology

A low and a high memory load was used as a basis of comparison. For the low memory load situation, the first 10 correlated patterns were associated together $(a - j)$. The patterns are 7×7 pixels images of alphabetic characters where a white pixel is given the value -1 and a black pixel the value 1. This led to a memory load equal to 20% (10/49) of the 49-dimensional space capacity. Normally, such a high load value cannot be handled by Kosko's BAM which is about 14%. For the high memory load situation, the associations were extended to all the 26 correlated patterns $(a - z)$. This last situation led to a memory load equal to

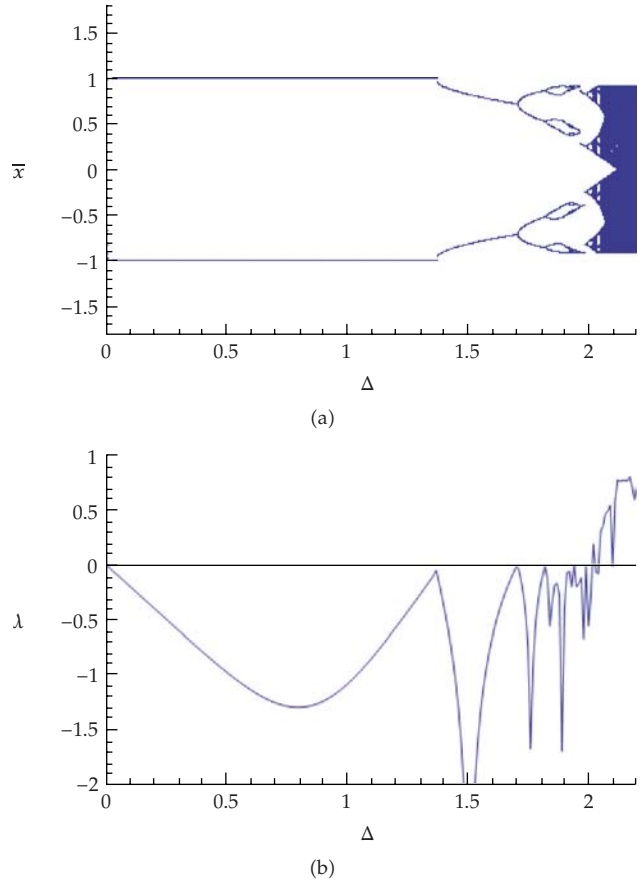


Figure 8: Bifurcation and Lyaponov exponent diagrams as a function of Δ.

53% (26/49). Usually, around 50% is about the maximum load an optimized Hebbian-type associative memory can stored without major performance decline [36]. Figure 9 illustrates the stimuli used for the simulation. The images were converted to vectors of 49 dimensions before being input to the network.

Both methods were tested with their output parameter set to 0.05 and 0.025. These values are much lower than the theoretical limit that was found analytically. The simulation results (not reported) show that a higher value makes the model unable to associate the patterns. The associations between patterns were accomplish using the learning rule (3.4) and (3.5) that will be described in the next section.

After associating the desired pattern pairs, the radiuses of attraction obtained by Euler and FORK were evaluated for noisy recall tasks. The first task consisted of recalling noisy patterns obtained by generating random normally distributed vectors that were added to the patterns. Each noisy pattern vector was distributed with a mean of 0 and a standard deviation of π (where π represents the desired proportion of noise). For the simulation, π varied from 0.1 to 1.0. The second task was to recall the correct associated stimulus from a noisy input obtained by randomly flipping pixels in the input pattern. The number of pixel flips varied from 0 to 10, thus corresponding to a noise proportion of 0 to 20%.

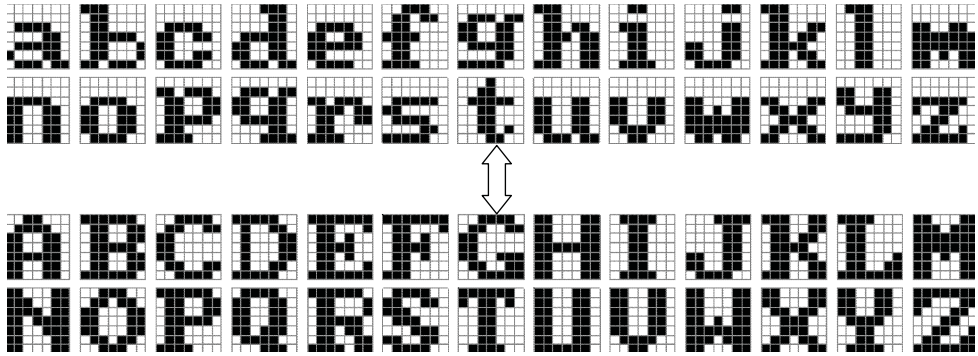


Figure 9: Pattern set used for numerical evaluations.

2.5.2. Results

Figure 10 illustrates the various performances. If the memory load is low (10 patterns over 49 pixels) as shown in the left column, then Euler's approximation has a slight advantage over FORK for random pixel flipping. This advantage is increased if the memory load is high (26 patterns over 49 pixels) as shown in the right column. Moreover, the higher the value of the output parameter, the higher the performance will be. However, if the output parameter is set to high, then the network might not converged to a fixed-point. Therefore, the choice for an output transmission will be based on Euler methods.

3. Model Description

3.1. Architecture

The proposed BAM architecture is similar to the one proposed by Hassoun [37] as illustrated in Figure 11, where $\mathbf{x}(0)$ and $\mathbf{y}(0)$ represent the initial input-states (stimuli); t is the number of iterations over the network; and $\mathbf{W}$ and $\mathbf{V}$ are weight matrices. The network is composed of two Hopfield-like neural networks interconnected in head-to-toe fashion. These interconnected layers allow a recurrent flow of information that is processed in a bidirectional way. The y-layer returns information to the x-layer and vice versa. If similar patterns are presented at both layers, then the network act like an autoassociative memory and if the patterns at the y-layer are associated with different ones in the x-layer, then the network acts like a heteroassociative memory [3]. As a result, it encompasses both unsupervised (self-supervised) and supervised learning. In this particular model, the two layers can be of different dimensions and contrary to usual BAM designs, the weight matrix from one side is not necessarily the transpose of that from the other side.

3.2. Transmission Function

Based, on the numerical results of the Section 2.5.2. , the transmission function is expressed by the following equations. The input activations to each of the y- and x-layers are computed

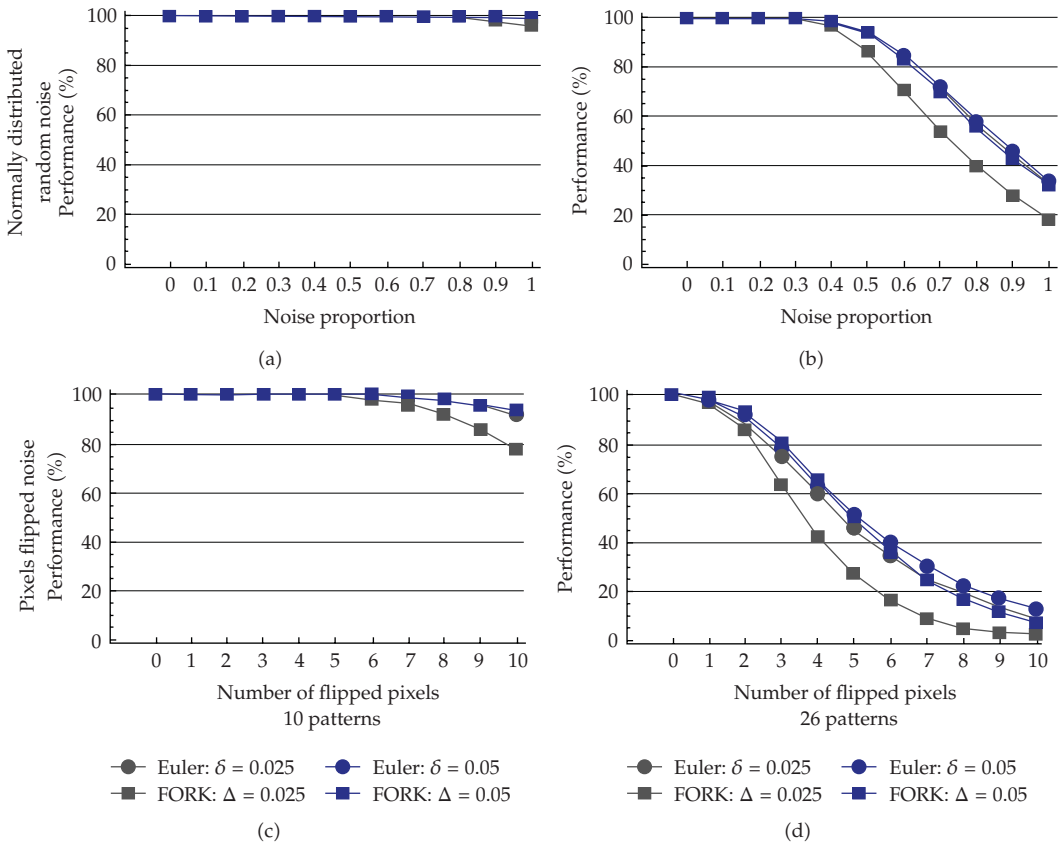


Figure 10: Performance comparison between Euler and FORK methods for different output parameters (0.05 and 0.025). Upper left panel low memory load under normally distributed noise Upper right high memory load under normally distributed noise. Lower left low memory load under pixels flipped noise. Lower right high memory load under pixels flipped noise.

as follows:

$$\begin{aligned} \mathbf{a}(t) &= \mathbf{W}\mathbf{x}(t), \\ \mathbf{b}(t) &= \mathbf{V}\mathbf{y}(t). \end{aligned} \tag{3.1}$$

The outputs are then obtained using Euler's approximation (2.12) defined by the following equations:

$$\begin{aligned} \forall i, \ldots, N, \quad \mathbf{y}_i(t+1) &= (1+\delta)\mathbf{a}_i(t) - \delta\mathbf{a}_i^3(t), \\ \forall i, \ldots, M, \quad \mathbf{x}_i(t+1) &= (1+\delta)\mathbf{b}_i(t) - \delta\mathbf{b}_i^3(t), \end{aligned} \tag{3.2}$$

where N and M are the number of units in each layer, i is the index of the respective vector element, $\mathbf{y}(t+1)$ and $\mathbf{x}(t+1)$ represent the layers' contents at time $t+1$, and δ is a general output parameter. The shape of this function is illustrated in Figure 6. This function has the

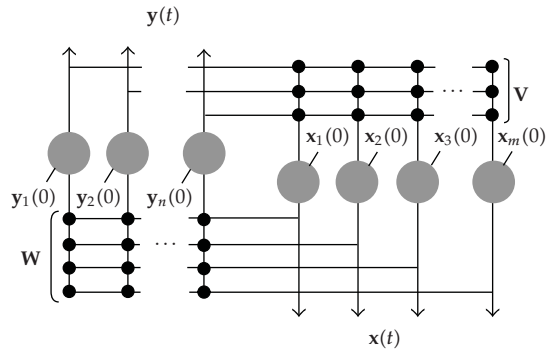


Figure 11: Architecture illustration of the bidirectional associative memory.

advantage of exhibiting continuous-valued (or gray-level) attractor behavior (for a detailed example, see [7]) when used in a recurrent network. Such properties contrast with standard nonlinear transmission functions, which only exhibit bipolar attractor behavior (e.g., [3]).

3.3. Learning

The network tries to solve the following nonlinear constraints:

$$\begin{aligned} \mathbf{X} &= f(\mathbf{VY}), \\ \mathbf{Y} &= f(\mathbf{WX}), \end{aligned} \tag{3.3}$$

where f is the transmission function defined previously (3.2). The form of the constraints and the recurrent nature of the underlying network call for a learning process that is executed online: the weights are modified in function of the input and the obtained output. In addition, since incremental learning is favored, the network must then be able to self-converge. As a result, the learning is based on the time difference Hebbian association [7, 38–40]. It is formally expressed by the following equations:

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta(\mathbf{y}(0) - \mathbf{y}(t))(\mathbf{x}(0) + \mathbf{x}(t))^T, \tag{3.4}$$

$$\mathbf{V}(k+1) = \mathbf{V}(k) + \eta(\mathbf{x}(0) - \mathbf{x}(t))(\mathbf{y}(0) + \mathbf{y}(t))^T, \tag{3.5}$$

where η represents the learning parameter. In (3.4) and (3.5), the weight updates follow this general procedure: first, initial inputs $\mathbf{x}(0)$ and $\mathbf{y}(0)$ are fed to the network, then, those inputs are iterated t times through the network (Figure 1). This results in outputs $\mathbf{x}(t)$ and $\mathbf{y}(t)$ that are used for the weight updates. Therefore, the weights will self-stabilize when the feedback is the same as the initial inputs ($\mathbf{y}(t) = \mathbf{y}(0)$ and $\mathbf{x}(t) = \mathbf{x}(0)$; in other words, when the network has developed fixed-points. This contrasts with most BAMs, where the learning is performed solely on the activation (offline). Learning convergence is a function of the value of the learning parameter η. For simplicity, we assumed that both input and output are the same ($\mathbf{y}(0) = \mathbf{x}(0)$). Therefore, to find the maximum value that the learning parameter can be

set to we need to find the derivation of learning equation when the slope is positive and then solve it for η [35]

$$\frac{\partial \mathbf{W}(k+1)}{\partial \mathbf{W}(k)} = \mathbf{W}(k) + \eta(\mathbf{x}(0) - \mathbf{x}(t))(\mathbf{x}(0) + \mathbf{x}(t))^T > 0. \tag{3.6}$$

If $t = 1$ and in the case of a one-dimensional pattern in a one-dimensional network, the situation simplifies to

$$\frac{\partial w(k+1)}{\partial w(k)} = w(k) + \eta\left(1 - (\delta+1)w(k) + \delta(w(k))^3\right)^2 > 0. \tag{3.7}$$

In this case, the network will converges when $w(k+1) = w(k) = 1$ and the solution will be

$$\eta < \frac{1}{2(1-2\delta)}, \quad \delta \neq \frac{1}{2}. \tag{3.8}$$

Just as any network, as the network increases in dimensionality, η must be set at lower values. Therefore, in the case of BAM of M and N dimensions, the learning parameter must be set according to

$$\eta < \frac{1}{2(1-2\delta)\,\mathrm{Max}[M,N]}, \quad \delta \neq \frac{1}{2}. \tag{3.9}$$

4. Simulations

The following simulations will first provide a numerical example to illustrate how the learning and recall are performed using a small network. The next simulation then reproduces a classic BAM one-to-one association's task. Finally, the third simulation extends the association property of the model to many-to-one association's task.

4.1. Numerical Example

The first simulation uses a toy example to show in details how the learning and recall processes are performed.

4.1.1. Learning

For this simulation the transmission function parameter (δ) was set to 0.1 and the learning parameter (η) was set to 0.01. Also, to limit the simulation time, the number of output iterations before each weight matrix update was set to $t = 1$ for all simulations. Learning

was carried out according to the following procedure:

Weight initialization at zero;

(1) random selection of a pattern pair,

(2) computation of $\mathbf{x}(t)$ and $\mathbf{y}(t)$ according to the transmission function (3.2),

(3) computation of the weight matrix ($\mathbf{W}$ and $\mathbf{V}$) update according to (3.4) and (3.5),

(4) repetition of steps (1) to (3) until the weight matrices converge.

The stimuli pairs for the simulation are given by

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}, \qquad \mathbf{Y} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{4.1}$$

First Learning Trial

Assume that the second pair is randomly selected. Then,

$$\mathbf{x}(0) = \begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix}^T, \qquad \mathbf{y}(0) = \begin{bmatrix} 1 & -1 & 1 \end{bmatrix}^T \tag{4.2}$$

and (3.2) yield zero-vectors for $\mathbf{y}(1)$ and $\mathbf{x}(1)$ since the weight connections are initially set at zero:

$$\mathbf{x}(1) = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T, \qquad \mathbf{y}(1) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T. \tag{4.3}$$

Using (3.4) and (3.5), the weight matrices are updated and their values are

$$\mathbf{W}(1) = \begin{bmatrix} 0.01 & 0.01 & 0.01 & -0.01 \\ -0.01 & -0.01 & -0.01 & 0.01 \\ 0.01 & 0.01 & 0.01 & -0.01 \end{bmatrix}, \qquad \mathbf{V}(1) = \begin{bmatrix} 0.01 & -0.01 & 0.01 \\ 0.01 & -0.01 & 0.01 \\ 0.01 & -0.01 & 0.01 \\ -0.01 & 0.01 & -0.01 \end{bmatrix}. \tag{4.4}$$

Second Learning Trial

Assume that the first pair is randomly selected. Therefore,

$$\mathbf{x}(0) = \begin{bmatrix} 1 & 1 & -1 & -1 \end{bmatrix}^T, \qquad \mathbf{y}(0) = \begin{bmatrix} 1 & -1 & -1 \end{bmatrix}^T. \tag{4.5}$$

Using (3.2) we compute $\mathbf{y}(1)$ and $\mathbf{x}(1)$ to get

$$\mathbf{x}(1) = \begin{bmatrix} 0.011 & 0.011 & 0.011 & -0.011 \end{bmatrix}^T, \qquad \mathbf{y}(1) = \begin{bmatrix} 0.022 & -0.022 & 0.022 \end{bmatrix}^T. \tag{4.6}$$

Using (3.4) and (3.5), the weight matrices are updated and their values are now

$$\mathbf{W}(2) = \begin{bmatrix} 0.0199 & 0.0199 & 0.0003 & -0.0199 \\ -0.0199 & -0.0199 & -0.0003 & 0.0199 \\ -0.0003 & -0.0003 & 0.02 & 0.0003 \end{bmatrix},$$
$$\mathbf{V}(2) = \begin{bmatrix} 0.02 & -0.02 & 0.0003 \\ 0.02 & -0.02 & 0.0003 \\ -0.0003 & 0.0003 & 0.0199 \\ -0.02 & 0.02 & -0.0003 \end{bmatrix}. \tag{4.7}$$

Notice that the weight matrices are not the transposed of each other like Kosko's [3].

k Learning Trial

If the process is iterated over and over, (e.g., 100 learning trials) the weight matrices converge to

$$\mathbf{W}(100) = \begin{bmatrix} 0.33 & 0.33 & 0 & -0.33 \\ -0.33 & -0.33 & 0 & 0.33 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{V}(100) = \begin{bmatrix} 0.5 & -0.5 & 0 \\ 0.5 & -0.5 & 0 \\ 0 & 0 & 1 \\ -0.5 & 0.5 & 0 \end{bmatrix}. \tag{4.8}$$

4.1.2. Recall

To test if the network can effectively recall a given pattern, a pattern is selected and it is iterated through the network until stabilization. For example, if the following pattern is presented to the network (noisy version of pattern 2) the different state-vectors will be:

$$\mathbf{x}(0) = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^T. \tag{4.9}$$

The output of the second layer will give

$$\mathbf{y}(1) = \begin{bmatrix} 0.363 & -0.363 & 1 \end{bmatrix}^T. \tag{4.10}$$

Then, this output is sent back to the first layer:

$$\mathbf{x}(1) = \begin{bmatrix} 0.344 & 0.344 & 1 & -0.344 \end{bmatrix}^T. \tag{4.11}$$

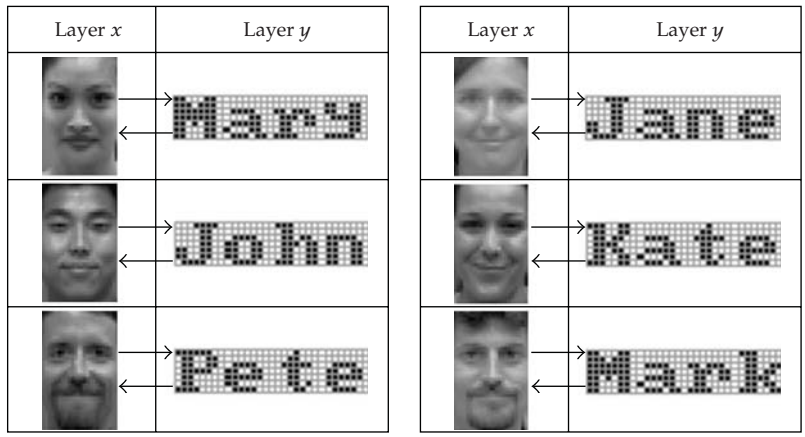


Figure 12: Patterns pairs used to trained the network.

The cycle will be repeated until convergence

$$\begin{aligned}
\mathbf{y}(2) &= \begin{bmatrix} 0.344 & -0.344 & 1 \end{bmatrix}^T, \\
\mathbf{x}(2) &= \begin{bmatrix} 0.395 & 0.395 & 1 & -0.395 \end{bmatrix}^T, \\
&\vdots \\
\mathbf{y}(50) &= \begin{bmatrix} 1 & -1 & 1 \end{bmatrix}^T, \\
\mathbf{x}(50) &= \begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix}^T.
\end{aligned} \tag{4.12}$$

Therefore, the new input is classified as part of the second pattern pair. The next section presents a classic example of one-to-one association.

4.2. One-to-One Association

The task of the network consists of associating a given picture with a name. Therefore, the network should output from a picture the corresponding name, and from the name the corresponding picture.

4.2.1. Methodology

The first stimulus set represents 8-bit grey-level pictures. Each image has a dimension of 38×24 and therefore forms a 912-dimensional real-valued vector. They are reduced size versions of the California Facial Expressions set (CAFE, [41]). Each pixel was normalized to values between -1 and 1. The second set consists of 4-letter words on a 7×31 grid that identify each picture. For each name a white pixel is assigned a value of -1 and a black pixel

Table 1: Network recall for a noisy input.

Original	Noisy version	Output ($k = 1$)		Output ($k = 2$)		Output ($k = 100$)	
		y	x	y	x	y	x
							

a value of +1. Each name forms a 217-dimensional bipolar vector. Therefore, the **W** weight matrix has 217×912 connections and the **V** weight matrix 912×217 connections. The network task was to associate each image with its corresponding name as depicted in Figure 12. The learning parameter (η) was set to 0.0025 and the output parameter(δ) to 0.1. Both values met the requirement for weight convergence and fixed-point development. Since the model's learning is online and iterative, the stimuli were not presented all at once. In order to save computational time, the number of iterations before each weight update was set to $t = 1$. The learning followed the same general procedure described in the previous simulation:

(1) initialization of weights to zero,

(2) random selection of a pair following a uniform distribution,

(3) stimuli iteration through the network according to Equations (3.1) and (3.2) (one cycle),

(4) weight update according to Equations (3.4) and (3.5),

(5) repetition of 2 to 4 until the desired number of learning trials is reached ($k = 15\,000$).

4.2.2. Results

It took about 12 000 learning trial before the learning converged. The network was able to perfectly associate a name with the corresponding picture, and vice versa. Table 1 to Table 4 illustrates some examples of the noise tolerance and pattern completion properties of the network. More precisely, Table 1 shows how the network was able to recall the proper name under a noisy input. The input consisted of the first picture contaminated with a noise level of 22%. In other words, 200 pixels (out of 912) were randomly selected and their values were multiplied by −1.

In addition, by the bidirectional nature of the network, it is also possible to not only recall the appropriate name but also to clean the noisy input. Table 2 shows an example of pattern completion. In this case the eye band consisted of pixels of zero value. The network was able to correctly output the name and restore the missing eyes. The network can also output a picture given an appropriate noisy name. For example, Table 3 shows that the network was able to recall the appropriate name even though the input name was incorrect (Kate instead of Katy). Finally, Table 4 shows that since "Pete" is the only name that begins with a "*P*", then only the first letter is necessary for the network to output the correct face. The general performance of the network has been evaluated by Chartier and Boukadoum [7]. The next section extends the simulations by introducing many-to-one associations.

Table 2: Network recall for an incomplete pattern.

Original	Noisy version	Output ($k = 1$)		Output ($k = 100$)	
		y	x	y	x
					

Table 3: Network recall for an incorrect name.

Original	Noisy version	Output ($k = 1$)		Output ($k = 2$)		Output ($k = 100$)	
		x	y	x	y	x	y
							

4.2.3. Transmission Function with Hard Limits

For some applications (e.g., [7]), It is desired that the transmission function lies within a fixed range of values. Saturating limits at −1 and 1 can be added to (3.2) and the transmission function is then expressed by the following equations:

$$\forall j, \ldots, N, \quad \mathbf{y}_j(t+1) = f\left(\mathbf{a}_j(t)\right) = \begin{cases} 1 & \text{if } \mathbf{a}_j(t) > 1 \\ -1 & \text{if } \mathbf{a}_j(t) > 1, \\ (\delta+1)\mathbf{a}_j(t) - \delta\mathbf{a}_j^3(t), & \text{else,} \end{cases}$$
$$\forall j, \ldots, M, \quad \mathbf{x}_j(t+1) = f\left(\mathbf{b}_j(t)\right) = \begin{cases} 1 & \text{if } \mathbf{b}_j(t) > 1 \\ -1 & \text{if } \mathbf{b}_j(t) > 1, \\ (\delta+1)\mathbf{b}_j(t) - \delta\mathbf{b}_j^3(t), & \text{else.} \end{cases} \tag{4.13}$$

In contrast to a sigmoid transmission function, this function is not asymptotic at the ±1 values, and it still has the advantage of exhibiting continuous-valued (or gray-level) attractor behavior when used in a recurrent network. Figure 13 illustrates the shape of the transmission function when $\delta = 0.5$.

We compared the same network with and without hard limits on the same task (Figure 9) previously described in Section 2.5.1. The performance of the network was evaluated with transmission function (δ) values between 0.05 to 0.3; values outside this range gave worst results. In addition, the network was compared with the best results ($\delta = 0.05$) obtained from the simulation illustrated in Figure 10 when no hard limits was used. After the association of the desired pattern pairs, radius of attraction of the network was evaluated

Table 4: Network recall for missing name parts.

Original	Noisy version	Output ($k = 1$)		Output ($k = 10$)		Output ($k = 100$)	
		x	y	x	y	x	y
							

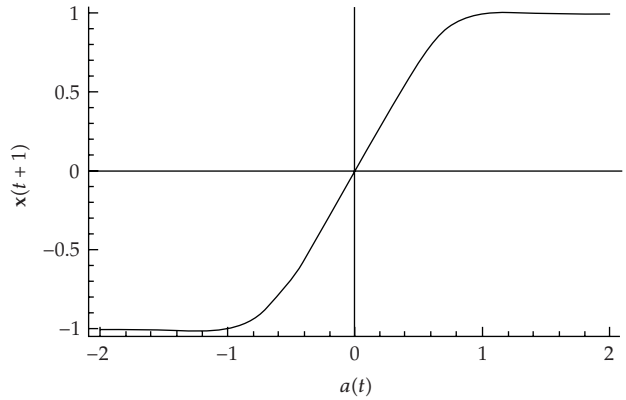


Figure 13: Transmission function when $\delta = 0.5$ with hard limits at −1 and +1.

under noisy input obtained by randomly flipping pixels in the input pattern. The number of pixel flips varied from 0 to 10.

Figure 14 illustrated the various performances. The results show that under low to medium noise, the network will have the best performance (about 10% increases) if no hard limits are used. However, under medium-to-high level of noise the situation is reverse; the network will have the best performance (about 5% increases) if hard limits are used ($\delta = 0.1$).

4.3. Many-to-One Association

This simulation illustrates many-to-one association. The idea is to associate different emotions depicted by different peoples to the proper name tag. Therefore, upon the presentation of a given images, the network should output the corresponding emotion. The simulation is based on Tabari et al. [42].

4.3.1. Methodology

As in the previous section, the first stimulus set represents 8-bits grey-level pictures. Each image has a dimension of 38 × 24 and therefore forms a 912-dimensional real value vector. They are reduced size version of the California Facial Expressions sample (CAFE, [41]). Each image pixels was rescaled to values between −1 and 1. For each of the 9 individuals 7 images reflect a given emotion (anger, disgust, fear, happy, maudlin, neutral, and surprised). The

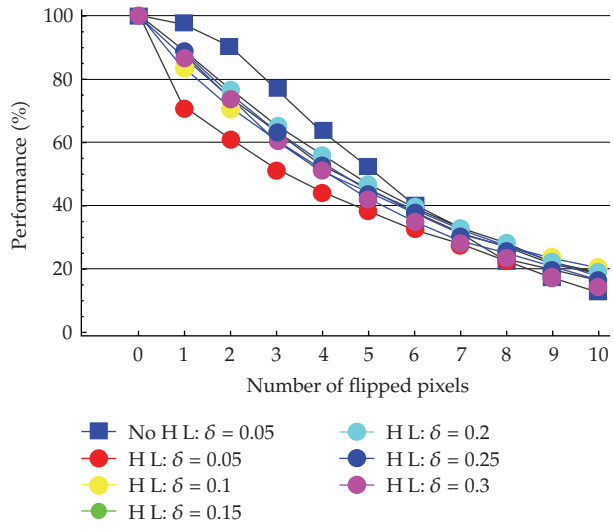


Figure 14: Comparison between the transmission function with (disks) or without hard limits (square) in function of various values of delta.

second set consists of letters place a 7×7 grid that identify each emotions (A, D, F, H, M, N, and S). For each letter a white pixel is assigned a value of -1 and a black pixel a value of $+1$. Each name forms a 49-dimensional bipolar vector. Therefore, the $\mathbf{W}$ weight matrix has 49×912 connections and the $\mathbf{V}$ weight matrix 912×49 connections. The network task was to associate each emotion, expressed by different person, with the corresponding letter (Figure 15). η was set to 0.0025 and δ to 0.1. Both values met the requirement for weight convergence and fixed-point behavior. The learning procedure followed the general procedure expressed previously.

4.3.2. Results

Once the learning trials were finished (about 1000 epochs), the network was able to correctly recall each emotion from the different people expressing it. As in one-to-one association, the network was able to removed the noise and correctly recall the appropriate emotion tag. Table 5 shows two examples of noisy images that were contaminated by 200-pixel flips (22%). Table 6 shows that the network recalled the appropriate emotion tag even in the absence of picture parts that are usually deemed important for identifying emotions, that is, the eyes and the mouth.

However, the most important property in this context is the fact that the two weight matrices are dynamically linked together. As was shown before, the network will output the corresponding letter given a face, but which face should the network output for a given letter? In this case, going from the letter layer to the picture layer implies one-to-many association. Without an architecture modification that can allow contextual encoding [7], the network will not be able to output the correct pictures. Rather, the network will average all the people emotional expression. Therefore, as shown in Figure 16, the network is able to extract what features in the images make each emotion different.

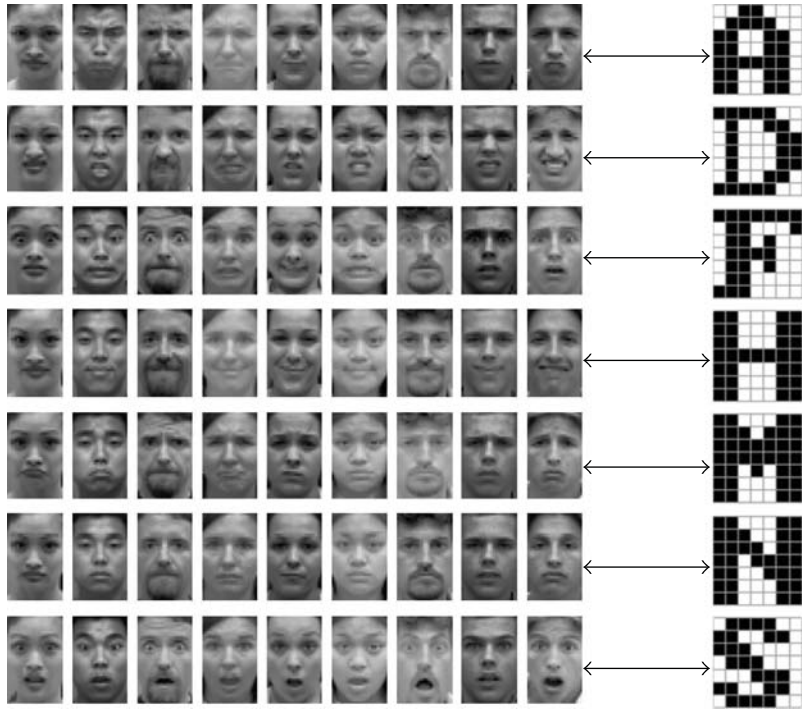


Figure 15: Patterns pairs used to trained the network.

Table 5: Network recall using a noisy input: 200 pixels flipped (22%).

Step 0 (Input)	Step 0 (Noisy input)	Step 1 (Output)
		
		

5. Model Extension to Temporal Associative Memory

Until now, the association between the different stimuli is static. In some situations, the network can also perform multistep pattern recognition [43]. Such is the case when the output is a time-series. To allow such encoding, the network's architecture is be modified.

Table 6: Network recall when important feature is removed (eyes and mouth).

Step 0 (Input)	Step 0 (Noisy input)	Step 1 (Output)
		
		

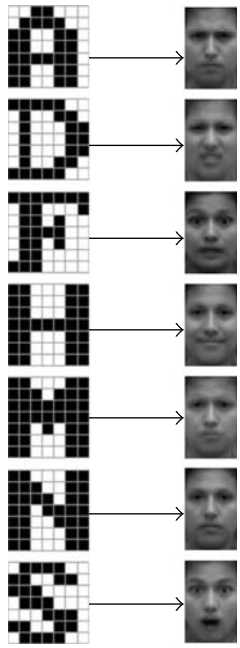


Figure 16: Pictures reconstructed using the emotion tags.

5.1. Architecture Modification

Figure 17 shows that, instead of having two heteroassociative networks connected together as in the Okajima et al. model [44], the network is composed of a heteroassociative and an autoassociative layer. The heteroassociative layer is used to map a given pattern to the next layer, whereas the autoassociative layer acts like a time delay circuit that feeds back the initial input to the heteroassociative layer. With this delay, it becomes possible for the overall network to learn temporal pattern sequences. Figure 17 illustrates the difference between a temporal associative memory and a bidirectional associative memory.

The initial value of the input pattern to the heteroassociative part is $\mathbf{y}(0)$, and its output, $\mathbf{y}(t)$, feeds the autoassociative part as $\mathbf{x}(0)$. The latter yields an output, $\mathbf{x}(t)$, that is

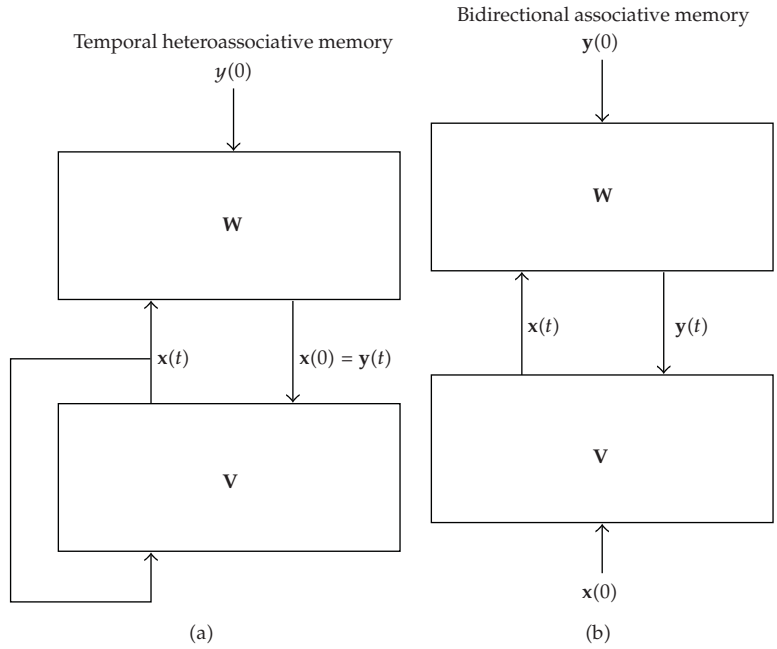


Figure 17: Comparison between the temporal associative memory and the standard BAM.

fed back into the heteroassociative part as well as into the autoassociative part. Thus, the autoassociative part serves as a context unit to the heteroassociative part. These two networks work in synergy to allow the necessary flow of information for, as our simulations will show, multistep pattern recall while still being able to filter noise out of the inputs.

5.2. Simplification of the Learning Function

In the case of autoassociative learning, the function expressed by (3.5) can be simplified by using the fact that the weight matrix is then square and symmetric. The symmetry property has the effect of canceling the cross terms in (3.5), and the autoassociative learning function becomes

$$\begin{aligned}\mathbf{V}(k+1) &= \mathbf{V}(k) + \eta\Big(\mathbf{x}(0)\mathbf{x}(0)^T + \mathbf{x}(0)\mathbf{x}(t)^T - \mathbf{x}(t)\mathbf{x}(0)^T - \mathbf{x}(t)\mathbf{x}(t)^T\Big),\\ \mathbf{V}(k+1) &= \mathbf{V}(k) + \eta\Big(\mathbf{x}(0)\mathbf{x}(0)^T - \mathbf{x}(t)\mathbf{x}(t)^T\Big),\end{aligned} \tag{5.1}$$

where $\mathbf{V}$ represents the connections weight matrix. The learning rule is then a sum of a positive and a negative Hebbian term [7, 38].

5.3. Simulation 1: Limit Cycle Behavior

This simulation illustrates how the network can perform temporal association. The task consists of learning different planar rotation of the same object.

Table 7: Five binary sequences composed of 8 time steps.

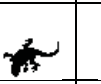
Sequence	45° →	90° →	135° →	180° →	225° →	270° →	315° →	0°
1								
2								
3								
4								
5								

Table 8: Network recall using a noisy input: 700 pixels flipped (28%).

Step 0 (Input)	Step 1 (Output)	Step 2 (Output)	Step 3 (Output)	Step 4 (Output)	Step 5 (Output)	Step 6 (Output)	Step 7 (Output)
	Step 8 (Output)	Step 9 (Output)	Step 10 (Output)	Step 11 (Output)	Step 12 (Output)	Step 13 (Output)	Step 14 (Output)

5.3.1. Methodology

Five different pattern sequences must be learned. Each sequence is composed of the same image with a 45 degrees planar rotation. Table 7 illustrates the five patterns sequences. The network has to associative each pattern of a sequence with the following one. In other words the network will associate the 45° with 90°, the 90° with 135°, the 135° with 180°, the 180° with the 225°, the 225° with 270°, the 270° with the 315°, the 315° with the 0°, and the 0° *with the 45°* image. Therefore, the steady-state of the network should be a limit cycle of period 8. Each binary stimulus was placed on a 50 × 50 grid, where white and black pixels were assigned −1 and 1 values, respectively. The free parameters were set to $\eta = 0.0002$ and $\delta = 0.1$, in accordance to the specification given in Chartier et al. [8].

5.3.2. Results

The network took about 2000 learning trials to converge and was able to correctly learn the 5 pattern sequences. It was also able to operate with noisy inputs and perform some generalization. For instance, Table 8 illustrates how the network was able to remove noise

Table 9: Network recall using a different picture: another example of butterfly.

Step 0 (Input)	Step 1 (Output)	Step 2 (Output)	Step 3 (Output)	Step 4 (Output)	Step 5 (Output)	Step 6 (Output)	Step 9 (Output)
							
	Step 8 (Output)	Step 9 (Output)	Step 10 (Output)	Step 11 (Output)	Step 12 (Output)	Step 13 (Output)	Step 14 (Output)
							

Table 10: Network recall using a different planar rotation: 275°.

Step 0 (Input)	Step 1 (Output)	Step 2 (Output)	Step 3 (Output)	Step 4 (Output)	Step 5 (Output)	Step 6 (Output)	Step 7 (Output)
							
	Step 8 (Output)	Step 9 (Output)	Step 10 (Output)	Step 11 (Output)	Step 12 (Output)	Step 13 (Output)	Step 14 (Output)
							

from the "fish" image as the temporal sequence was recalled. Table 9 shows that the network can generalize its learning to use similar objects. In this case a new "butterfly" picture was used as the initial input. As the stimulus iterates through the network, the network recalled the original "butterfly" depicted in Table 7 (the butterfly output at step 8 is different from the original one at step 0). Finally, Table 10 shows that the network can correctly output the image sequence under small planar rotation variations of the initial input image. In this particular example the new "dinosaur" picture represents a 275° rotation (270° + 5°). Although that particular rotation was not part of initial sequence set, the network was able to correctly recall the appropriate picture sequence.

5.4. Simulation 2: Fixed-Point Behavior

Again, this simulation illustrates how the network can perform temporal association. The task is to learn different planar rotations of the same object. Contrarily to the previous simulation, once the network has been through every planar rotation it converges to a fixed-point.

Table 11: Five binary sequences composed of 8 time steps.

Sequence	45° →	90° →	135° →	180° →	225° →	270° →	315° →	0° ↻
1								
2								
3								
4								
5								

Table 12: Network recall using a noisy input: 700 pixels flipped (28%).

Step 0 (Input)	Step 1 (Output)	Step 2 (Output)	Step 3 (Output)	Step 4 (Output)	Step 5 (Output)	Step 6 (Output)	Step 7 (Output)

5.4.1. Methodology

Five different pattern sequences must be learned. Each sequence is composed of the same image with a 45° planar rotation. Table 11 illustrates the five patterns sequences. The network must associative each pattern of a sequence with the following one. In other words the network will associate the 45° with 90°, the 90° with 135°, the 135° with 180°, the 180° with the 225°, the 225° with 270°, the 270° with the 315°, the 315° with the 0° and *the 0° with the 0°* Image. This last association will creates a fixed-point attractor that can then be used for other types of association in more complex architectures as will be shown in the next section. Each binary stimulus was placed on a 50×50 grid, where white and black pixels were assigned a -1 and 1 value, respectively. The free parameters were set to $\eta = 0.0002$ and $\delta = 0.1$, in accordance to the specification given in [8].

5.4.2. Results

The network took about 2000 learning trials before convergence occurred, and it was able to correctly learn the 5 pattern sequences. The same examples for noise tolerance and generalization were used to compare the network performance between the limit cycle and the fixed-point condition. Table 12 shows that the network can eliminate noise as the sequence is recalled. Table 13 shows learning generalization to other similar objects. After 9 time steps, the network recalled the "butterfly" that is depicted in Table 11. Finally, Table 14 shows that the network can also output the correct image sequence under small initial variations of

Table 13: Network recall using a different picture: another example of butterfly.

Step 0 (Input)	Step 1 (Output)	Step 2 (Output)	Step 3 (Output)	Step 4 (Output)	Step 5 (Output)	Step 6 (Output)	Step 9 (Output)
							

Table 14: Network recall using a different planar rotation: 275°.

Step 0 (Input)	Step 1 (Output)	Step 2 (Output)	Step 3 (Output)	Step 4 (Output)	Step 5 (Output)	Step 6 (Output)	Step 7 (Output)
							

planar rotation. In this particular example the "dinosaur" picture represent a 275° rotation as in the previous section.

6. Multidirectional Associative Memories

Sometimes association must be generalized to more than two directions. To deal with multiple associations, Hagiwara [45] proposed a Multidirectional Associative Memory [MAM]. The architecture was later extended to deal with temporal sequences [46]. This section shows some properties of a MAM that is composed of one temporal heteroassociative memory and one bidirectional associative memory. As in the previous simulations, only the network topology is modified as Figure 18 shows; the learning and transmission function remain the same. In this simulation the information received by the y-layer at time t consist of $\mathbf{x}(t)$ and $\mathbf{z}(t)$. The feedback sent by $\mathbf{z}(t)$ is useful only once the network reaches steady-state, the last pattern of the stimuli set. Therefore, to minimize its effect during the temporal recall, the feedback value of $\mathbf{z}(t)$ is lower than $\mathbf{x}(t)$. This is formally expressed by

$$y(t) = f(\mathbf{W}1\mathbf{x}(t) + \alpha\mathbf{W}2z(t)), \tag{6.1}$$

where $0 < \alpha \ll 1$, $\mathbf{W1}$ and $\mathbf{W2}$ represent the weight connections linking $\mathbf{x}(t)$ and $\mathbf{z}(t)$, respectively. Taken together, $\mathbf{W}1$ and $\mathbf{W}2$ form the $\mathbf{W}$ weight matrix.

6.1. Methodology

The simulation performs the multistep pattern recognition described in the previous section in combination with a standard one-to-one association. The one-to-one association is performed only when the network is at a given fixed-point. In other words the association will occur only when the output is at a 0° planar rotation for the recalled sequence (Table 11). The association is made between the given picture and the first letter of its name as illustrated in Figure 19. The learning and transmission function parameters were set to $\eta = 0.0001$ and $\delta = 0.1$. The z-layer feedback parameter was set to $\alpha = 0.4$.

Table 15: Multistep recall of the "runner" binary picture (45° planar rotation).

Step	Input	Out 1	Out 2	Out 1 + α Out 2
1				
2				
3				
4				
5				
6				
7				
8				
9				

6.2. Results

Of course, if the temporal or bidirectional associative sections are independently activated, the resulting network performance will be the same as the previous sections. What matter in this case is the output state of the network $\mathbf{y}(t)$. The best case is when a pattern is presented to the network that is closed to its fixed-point behavior (315°). The effect of the feedback coming from z-layer will be limited. On the opposite, the worst case is when a pattern is presented with a 45° planar rotation. In this case, it will need 8 time steps before the output vector is at its fixed-point behavior. From this point on, the picture identification process can occur. If the feedback coming from the z-layer is too strong, it could deviate the output vector's

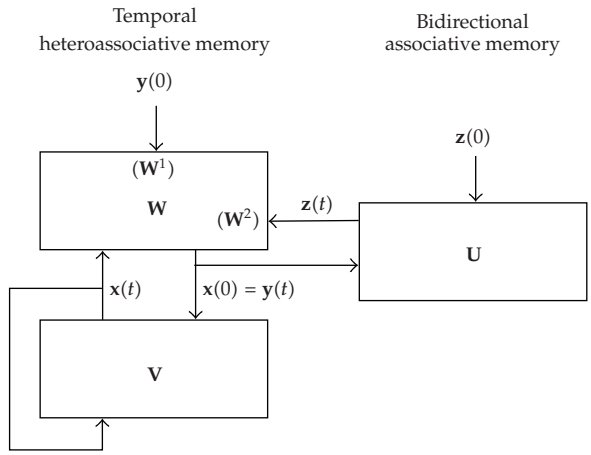


Figure 18: Architecture of the multidirectional associative memory. The network is composed of a temporal heteroassociative memory and a bidirectional associative memory.

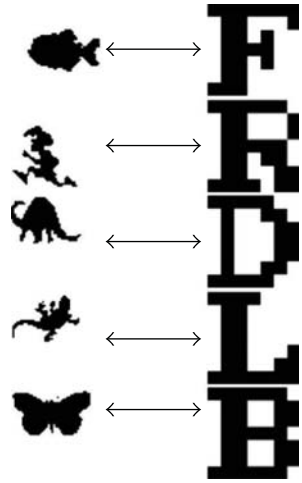


Figure 19: Patterns pairs used to trained the network.

trajectory. This could make the network converge to the wrong fixed-point and lead to an incorrect name tag. Because of this, the feedback value of the z-layer must be low.

Table 15 shows an example of the "runner" given an initial 45° planar rotation. After one time step the network correctly output the 90° picture (Out 1). Of course, the name tag is incorrect (Out 2). Now the new input will correspond to the output of the x layer (Out 10) and the feedback given by the z-layer (Out 2 not shown). The overall effect will be Out 1 + α Out 2 and will be used as the new input. The process is repeated until convergence. After 9 time steps, Table 15 shows that the correct 0° "runner" picture is output as well as the correct name tag letter "R".

7. Conclusion

Several interesting properties of a BAM architecture have been shown. First, it was shown that a simple time-delay Hebbian learning can perform one-to-one association and many-to-one association with both binary and real-valued pattern. In addition, the BAM can be modified in such a way that both heteroassociation and autoassociation can be accomplished within the same architecture. In this case, the autoassociative part act like a time delay and the overall network can be used for temporal association. A simple case of one-step delay was shown. However, by adding l extra layers of autoassociation, the network can be easily modified to handle l-step delays. Finally, if a combination of temporal and bidirectional associations is grouped into a multidirectional associative memory more complex behaviors can be obtained.

In all cases, the same learning and transmission function are used. The only modification concerned the network topology. This property gives the network a high internal consistency. More complex architectures are possible, but a really powerful implementation improvement would be to develop an algorithm that guides the architecture growth based on the problem to be solved. Therefore, the architecture could be modified in function of the task to be performed and the desired behaviors, using BAMs as building blocks.

Acknowledgment

This research was supported in part by Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, no. 8, pp. 2554–2558, 1982.

[2] J. A. Anderson, J. W. Silverstein, S. A. Ritz, and R. S. Jones, "Distinctive features, categorical perception, and probability learning: some applications of a neural model," *Psychological Review*, vol. 84, no. 5, pp. 413–451, 1977.

[3] B. Kosko, "Bidirectional associative memories," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 18, no. 1, pp. 49–60, 1988.

[4] T. Kohonen, "Correlation matrix memories," *IEEE Transactions on Computers*, vol. 21, no. 4, pp. 353–359, 1972.

[5] M. E. Acevedo-Mosqueda, C. Yáñez-Márquez, and I. López-Yáñez, "Alpha-Beta bidirectional associative memories: theory and applications," *Neural Processing Letters*, vol. 26, no. 1, pp. 1–40, 2007.

[6] S. Arik, "Global asymptotic stability analysis of bidirectional associative memory neural networks with time delays," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 580–586, 2005.

[7] S. Chartier and M. Boukadoum, "A bidirectional heteroassociative memory for binary and grey-level patterns," *IEEE Transactions on Neural Networks*, vol. 17, no. 2, pp. 385–396, 2006.

[8] S. Chartier, P. Renaud, and M. Boukadoum, "A nonlinear dynamic artificial neural network model of memory," *New Ideas in Psychology*, vol. 26, no. 2, pp. 252–277, 2008.

[9] S. Du, Z. Chen, Z. Yuan, and X. Zhang, "Sensitivity to noise in bidirectional associative memory (BAM)," *IEEE Transactions on Neural Networks*, vol. 16, no. 4, pp. 887–898, 2005.

[10] T. D. Eom, C. Choi, and J. J. Lee, "Generalized asymmetrical bidirectional associative memory for multiple association," *Applied Mathematics and Computation*, vol. 127, no. 2-3, pp. 221–233, 2002.

[11] H. Oh and S. C. Kothari, "Adaptation of the relaxation method for learning in bidirectional associative memory," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 576–583, 1994.

[12] C.-S. Leung, "Optimum learning for bidirectional associative memory in the sense of capacity," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 5, pp. 791–796, 1994.
[13] D. Shen and J. B. Cruz Jr., "Encoding strategy for maximum noise tolerance bidirectional associative memory," *IEEE Transactions on Neural Networks*, vol. 16, no. 2, pp. 293–300, 2005.
[14] H. Shi, Y. Zhao, and X. Zhuang, "A general model for bidirectional associative memories," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 28, no. 4, pp. 511–519, 1998.
[15] Y. F. Wang, J. B. Cruz Jr., and J. H. Mulligan Jr., "Two coding strategies for bidirectional associative memory," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 81–92, 1990.
[16] T. Wang, X. Zhuang, and X. Xing, "Weighted learning of bidirectional associative memories by global minimization," *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 1010–1018, 1992.
[17] L. Wang and X. Zou, "Capacity of stable periodic solutions in discrete-time bidirectional associative memory neural networks," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 51, no. 6, pp. 315–319, 2004.
[18] Y. Wu and D. A. Pados, "A feedforward bidirectional associative memory," *IEEE Transactions on Neural Networks*, vol. 11, no. 4, pp. 859–866, 2000.
[19] Z. B. Xu, Y. Leung, and X. W. He, "Asymmetric bidirectional associative memories," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 10, pp. 1558–1564, 1994.
[20] X. Zhuang, Y. Huang, and S. Chen, "Better learning for bidirectional associative memory," *Neural Networks*, vol. 6, no. 8, pp. 1131–1146, 1993.
[21] G. Costantini, D. Casali, and R. Perfetti, "Neural associative memory storing gray-coded gray-scale images," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 703–707, 2003.
[22] C.-C. Wang, S.-M. Hwang, and J.-P. Lee, "Capacity analysis of the asymptotically stable multi-valued exponential bidirectional associative memory," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 26, no. 5, pp. 733–743, 1996.
[23] D. Zhang and S. Chen, "A novel multi-valued BAM model with improved error-correcting capability," *Journal of Electronics*, vol. 20, no. 3, pp. 220–223, 2003.
[24] J. M. Zurada, I. Cloete, and E. van der Poel, "Generalized Hopfield networks for associative memories with multi-valued stable states," *Neurocomputing*, vol. 13, no. 2–4, pp. 135–149, 1996.
[25] L. Wang, "Multi-associative neural networks and their applications to learning and retrieving complex spatio-temporal sequences," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 29, no. 1, pp. 73–82, 1999.
[26] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.
[27] H. Wakuya and J. M. Zurada, "Bi-directional computing architecture for time series prediction," *Neural Networks*, vol. 14, no. 9, pp. 1307–1321, 2001.
[28] G. Bradski, G. A. Carpenter, and S. Grossberg, "STORE working memory networks for storage and recall of arbitrary temporal sequences," *Biological Cybernetics*, vol. 71, no. 6, pp. 469–480, 1994.
[29] B. B. Nasution and A. I. Khan, "A hierarchical graph neuron scheme for real-time pattern recognition," *IEEE Transactions on Neural Networks*, vol. 19, no. 2, pp. 212–229, 2008.
[30] J. A. Starzyk and H. He, "Anticipation-based temporal sequences learning in hierarchical structure," *IEEE Transactions on Neural Networks*, vol. 18, no. 2, pp. 344–358, 2007.
[31] D.-L. Lee, "A discrete sequential bidierectional associative memory for multi step pattern recognition," *Pattern Recognition*, vol. 19, pp. 1087–1102, 1988.
[32] L. Wang, "Heteroassociations of spatio-temporal sequences with the bidirectional associative memory," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, pp. 1503–1505, 2000.
[33] R. W. Zhou and C. Quek, "DCBAM: a discrete chainable bidirectional associative memory," *Pattern Recognition Letters*, vol. 17, no. 9, pp. 985–999, 1996.
[34] A. A. Koronovskii, D. I. Trubetskov, and A. E. Khramov, "Population dynamics as a process obeying the nonlinear diffusion equation," *Doklady Earth Sciences*, vol. 372, pp. 755–758, 2000.
[35] D. Kaplan and L. Glass, *Understanding Nonlinear Dynamics*, Springer, New York, NY, USA, 1995.
[36] L. Personnaz, I. Guyon, and G. Dreyfus, "Information storage and retrieval in spin-glass like neural networks," *Journal of Physics Letters*, vol. 46, pp. L359–L365, 1985.
[37] M. H. Hassoun, "Dynamic heteroassociative neural memories," *Neural Networks*, vol. 2, no. 4, pp. 275–287, 1989.
[38] S. Chartier and R. Proulx, "NDRAM: nonlinear dynamic recurrent associative memory for learning bipolar and nonbipolar correlated patterns," *IEEE Transactions on Neural Networks*, vol. 16, no. 6, pp. 1393–1400, 2005.
[39] B. Kosko, "Unsupervised learning in noise," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 44–57, 1990.

[40] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1988.
[41] M. N. Dailey, G. W. Cottrel, and J. Reilly, HYPERLINK, 2001, http://www.cs.ucsd.edu/users/gary/CAFE/.
[42] K. Tabari, M. Boukadoum, S. Chartier, and H. Lounis, "Reconnaissance d'expressions faciales à l'aide d'une mémoire associative bidirectionnelle à fonction de sortie chaotique," in *Proceedings of Maghrebian Conference on Software Engineering and Artificial Intelligence*, pp. 422–426, Agadir, Morocco, December 2006.
[43] S. Chartier and M. Boukadoum, "A sequential dynamic heteroassociative memory for multistep pattern recognition and one-to-many association," *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 59–68, 2006.
[44] K. Okajima, S. Tanaka, and S. Fujiwara, "A heteroassociative memory network with feedback connection," in *Proceedings of the 1st International Joint Conference on Neural Networks*, pp. H.711–H.7188, San Diego, Calif, USA, 1987.
[45] M. Hagiwara, "Multidirectional associative memory," in *Proceeding of the International Joint Conference on Neural Networks*, pp. 3–6, Washington, DC, USA, 1990.
[46] A. F. R. Araujo and M. Vieira, "Temporal multidirectional associative memory: adaptable, continuous, and self-connected MAM," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 2, pp. 1194–1198, Houston, Tex, USA, June 1997.

Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2011, Article ID 458768, 22 pages
doi:10.1155/2011/458768

Research Article

Hysteresis Nonlinearity Identification Using New Preisach Model-Based Artificial Neural Network Approach

Mohammad Reza Zakerzadeh,[1] Mohsen Firouzi,[2] Hassan Sayyaadi,[1] and Saeed Bagheri Shouraki[2]

[1] *School of Mechanical Engineering, Sharif University of Technology, P.O. Box 11155-9567, Tehran, Iran*
[2] *Electrical Engineering Department, Artificial Creature Lab, Sharif University of Technology, P.O. Box 11155-9567, Tehran, Iran*

Correspondence should be addressed to Mohammad Reza Zakerzadeh,
mzakerzadeh@mech.sharif.edu

Received 29 September 2010; Revised 26 December 2010; Accepted 15 February 2011

Academic Editor: I. Stamova

Preisach model is a well-known hysteresis identification method in which the hysteresis is modeled by linear combination of hysteresis operators. Although Preisach model describes the main features of system with hysteresis behavior, due to its rigorous numerical nature, it is not convenient to use in real-time control applications. Here a novel neural network approach based on the Preisach model is addressed, provides accurate hysteresis nonlinearity modeling in comparison with the classical Preisach model and can be used for many applications such as hysteresis nonlinearity control and identification in SMA and Piezo actuators and performance evaluation in some physical systems such as magnetic materials. To evaluate the proposed approach, an experimental apparatus consisting one-dimensional flexible aluminum beam actuated with an SMA wire is used. It is shown that the proposed ANN-based Preisach model can identify hysteresis nonlinearity more accurately than the classical one. It also has powerful ability to precisely predict the higher-order hysteresis minor loops behavior even though only the first-order reversal data are in use. It is also shown that to get the same precise results in the classical Preisach model, many more data should be used, and this directly increases the experimental cost.

1. Introduction

Today hysteresis modeling is one of the most interesting and challenging field of study in many engineering applications such as shape memory alloy (SMA), piezoelectric, piezocermaic, magnetostrictive, and electromechanical actuators. Since unmodeled hysteresis causes inaccuracy in trajectory tracking and decreases the performance of control systems,

an accurate modeling of hysteresis behavior for performance evaluation and identification as well as controller design is essentially needed. To overcome this drawback, it is necessary to develop hysteresis models that not only their parameters can easily and precisely be identified but also are suitable for real-time control and compensation system design [1].

Two different methods of modeling have been proposed to capture the observed hysteretic characteristics [2]. The first group of models is derived from the underlying physics of hysteresis and combined with empirical factors to describe the observed characteristics [3–5]. However, these models have limited applicability, as the physical basis of some of the hysteresis characteristics is not completely understood [6]. Furthermore, considerable effort is required in identifying and tuning the model parameters to accurately describe the hysteresis nonlinearity. Another major drawback of these physical models is that they are specific to a particular type of system, and this implies separate controller design techniques for each system [7].

The second group of models is based on the phenomenological nature and mathematically describes the observed phenomenon without necessarily providing physical insight into the problems [8–14]. Among these models, the Preisach model has found extensive application for modeling hysteresis in SMAs and other smart actuators [8, 10, 14]. Although the Preisach model does not provide physical insight into the problem, it provides a means of developing phenomenological model that is capable of predicting behaviors similar to those of the physical systems. Therefore, it is a convenient tool for hysteresis identification and compensation [13].

In Preisach modeling technique overall system with hysteresis behavior is modeled by weighted parallel connections of nonideal relays termed as Preisach elemental operators, $\gamma_{\alpha,\beta}$ (Figures 1 and 2). Every elemental operator with output +1 or −1 (zero in some models) as a nonlinear operator consists of two parameters, α, β, which denote upper and lower switching values of input, respectively. Along with the set of operators $\gamma_{\alpha,\beta}$ is an arbitrary weight function $\mu(\alpha,\beta)$, called the Preisach density function (PDF), which works as a local influence of each operator in overall hysteresis model.

There are two general approaches to implement the Preisach model. The first approach is trying to approximate the PDF by some predefined special forms with a few undetermined parameters. Each material has an optimal distribution function and parameter, which deliver the best result with respect to the experimental data [15]. In the identification process of this approach, the unknown parameters are determined in order to minimize the error between the output of the model and experimental data by numerical curve fitting algorithms such as minimum least square method. In other words, optimal fitting of the calculated outputs to measured data of the real system can determine the values of parameters used in the assumed function.

The main disadvantage of this approach is the fact that the accuracy of the model is strongly dependent on the type of the candidate function and the number of its parameters. Moreover, it is not easy to determine the suitable shape of the distribution functions by less experimental data. Also, to simulate the hysteresis loops of different materials, the different distribution functions and parameters are needed. In order to overcome this drawback, a numerical density function approximation method based on mapping Preisach model into a linear equation system was proposed by Shirley and Venkatraman [16]. Another solution is identifying the Preisach function by using artificial neural networks (ANNs) or fuzzy approximators. As a matter of fact, the neural networks or fuzzy engines provide the parameters necessary for describing a given hysteresis loop, under the assumption that the type of the Preisach function is already known. In [17], the identification of the Preisach

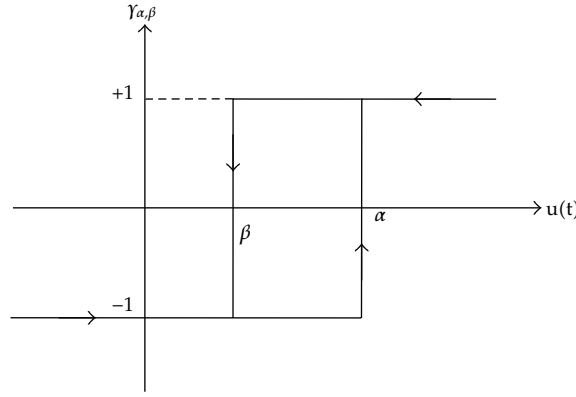


Figure 1: Preisach elemental operator.

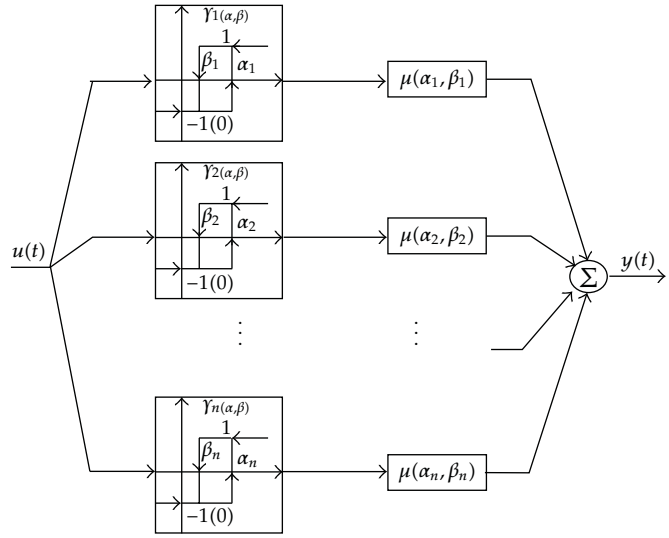


Figure 2: Block representation of the classical Preisach model.

function of a material is performed by using a neural network trained by a collection of hysteresis curves, whose Preisach functions are known. When a new hysteresis curve is given as input to this neural network, it is able to give as output both the functional dependence of the Preisach function and its numerical parameters. Moreover, the proposed method allows to determine any usual analytical structure of the Preisach function provided that a suitable training set is adopted. The method has been applied to the prediction of hysteresis loops of magnetic sheets by using Preisach method and it has shown a good numerical accuracy. In [18], two different identification techniques have been proposed. In this paper, hysteresis is modeled by applying the classical Preisach model whose identification procedure is performed by the adoption of both a fuzzy approximator and a feed-forward neural network to analytically reconstruct the Preisach distribution function, without any special smoothing of the measured data, owing to the filtering capabilities of the neurofuzzy interpolators.

Since, in the first form of Preisach model, the numerical evaluation of double integral is a time consuming process that hinders the practical application of the Preisach model, in the second approach, a numerical evaluation of double integral instead of density function approximation developed by Mayergoyz was proposed [14]. Despite that the Mayergoyz approach has received general acceptance to capture the main features of hysteresis phenomena, it is not suitable for real-time control applications. There have been some limitations on the accuracy of this method in addition to its computation time for the model inversion. These are caused by the restrictions of switching points in the Preisach plane, the inaccuracy of data measurement, and the geometrical interpolation error appeared in the identification process of classical Preisach model [13].

Several extensions and modifications of the numerical classical Preisach model using fuzzy inference engine, artificial neural networks as well as neurofuzzy identifier have been presented in order to remedy these problems [6, 19, 20]. However, it is known that these tools can only be available for the approximation of the continuous systems with one-to one and multi-to-one mappings and they are unable to directly model the systems with multi-valued mapping such as hysteresis [21]. Therefore, they are needed to utilize the local memory property of the classical Preisach model.

Ahn and Kha [6] presented a new modification of the numerical classical Preisach based on geometrical implementation model using a fuzzy inference engine. The experimental evaluation showed that the model is suitable in predicting the hysteresis phenomenon of the SMA actuators. They also used the fuzziness based inverse Preisach model incorporated in a closed loop internal model control to investigate the control performance and the effect of hysteresis compensation for SMA actuators. In spite of strong power modeling of fuzzy inference engines (FIE) through a simple linguistic computation paradigm, these tools suffer from a lack of learning algorithms to adjust the best membership functions of input domains. But, on the other hand, the artificial neural networks have a good ability to map input-output patterns through straightforward learning algorithms. To overcome the aforementioned problems of FIEs and utilizing the excellent learning capability of neural networks, neurofuzzy tools are developed in the field of computational intelligent [22]. Dlala and Arkkio [19] proposed a method that identified the numerical classical Preisach model by using a neurofuzzy approximator instead of interpolation which is used in the Mayergoyz approach. This novel method utilizes the available data in the major loop in the identification process and omits the need of measuring the first-order reversal curves. They also applied their method to predict cyclic minor loops of a soft magnetic composite and verified the accuracy of proposed method with respect to experimental data. Since the adaptive learning process of neurofuzzy approach is much time, consuming, there is a limitation on the number of fuzzy rules. Therefore, often there is a trade-off between the system accuracy, computation run time and fuzzy rules number [20].

ANNs have powerful fault tolerant computing ability which has been used to model a wide range of systems for mathematical models which either cannot be defined or are ill-defined. Similarly, ANNs are well suited for systems that involve complex, multivariable processes. One of the advantages of using neural networks for hysteresis modeling is that their parameters can be updated online to track the change of the environment or operating condition. It is demonstrated in this paper that ANNs are individually capable of modeling hysteresis based on classical Preisach approaches without suffering the mentioned problems of fuzzy inference engines as well as neurofuzzy systems. To evaluate this approach in hysteresis modeling, a one-dimensional flexible aluminum beam, whose deflection is controlled by an SMA wire as an actuator, is used. Experimental results show the power

of the proposed ANN based Preisach hysteresis model in comparison with classical Preisach model and mentioned Shirley et al. approach.

The paper is organized as follows. Section 2 is dedicated to the classical Preisach model and its geometrical interpretation. In this section an introduction of the classical Preisach model is presented. The Numerical Preisach model is discussed in more detail in Section 3 and its implementation method is discussed in Section 4. Then, in Section 5 a method developed by Shirley et al. for PDF approximation is presented. In Section 6 the structure of proposed novel ANN based Preisach model is described. The evaluation of the presented model is compared with experimental results in Section 7. Finally, the concluding comments are provided in Section 8.

2. Classical Preisach

Preisach model is a famous hysteresis identification technique, which is first introduced on the base of phenomenological analysis of ferromagnetic materials by German physicist F. Preisach almost 75 years ago [23]. The Russian mathematician, Krasnoselskii, in 1970 represented Preisach model into a pure formulized mathematical form in which hysteresis is modeled by linear combination of hysteresis operators [24]. Mathematical form of the classical Preisach model can be sketched by equation:

$$f(t) = \iint_{\alpha \geq \beta} \mu(\alpha, \beta)\ \gamma_{\alpha,\beta}[u(t)]\, d\alpha\, d\beta, \tag{2.1}$$

where $f(t)$ is the output of the model at state t and $u(t)$ is the input at the same state, and $\gamma_{\alpha,\beta}$ denotes elementary hysteresis operator with α and β ($\alpha \geq \beta$) parameters as upper and lower switching values, respectively (see Figure 1). Output of elemental operators would be only +1 or −1 (zero in some models). In (2.1), $\mu(\alpha, \beta)$ is density function value or Preisach function corresponding to α and β which should be determined by use of some experimental data. This distributed weighting describes the relative contribution of each relay in overall hysteresis system (see Figure 2). Equation (2.1) can be represented in summation form of finite number of rectangular elemental Preisach operators $\gamma_{\alpha_k,\beta_k}$ as

$$f(t) = \sum_{j=1}^{N} \sum_{i=1}^{N} \mu(\alpha_i, \beta_j) \gamma_{\alpha_i \beta_j}[u(t)], \tag{2.2}$$

in which

$$\alpha_i = \beta_i = \alpha_1 - 2\frac{(i-1)}{N-1}\alpha_1,$$

$$\gamma_{\alpha\beta}[u(t)] = \begin{cases} -1, & u(t) \leq \beta, \\ 1, & u(t) \geq \alpha, \\ \text{maintain}, & \alpha \leq u(t) \leq \beta, \end{cases} \tag{2.3}$$

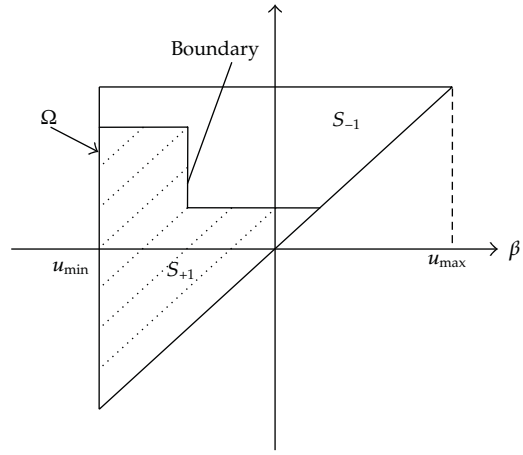


Figure 3: Geometrical representation of Preisach (α-β) plane.

where the approximation of dual integral form of classical Preisach model is represented by a dual sigma. Assume that normalized symmetrical hysteresis form α_1 is up saturation point and $-\alpha_1$ is down saturation point.

For better representation of Preisach model, let us define a plane with α and β coordinates where each elemental operator is denoted by a point (β, α) with a specific weighting value (Figure 3). PDF weighting values outside of Preisach triangle (Ω) must be zero, and also $u_{\max}$ and $u_{\min}$ denote upper and lower saturation points of input, respectively. When input is monotonically decreased or increased through $\alpha = \beta$ line in α-β plane, boundary line divides α-β plane to S_{-1} and S_{+1} area.

Dashed area in Figure 3 denoted by S_{+1} consists of points $\{\alpha_k, \beta_k \mid \gamma_{\alpha_k,\beta_k} = +1\}$, and blank area denoted by S_{-1} consists of points $\{\alpha_k, \beta_k \mid \gamma_{\alpha_k,\beta_k} = -1\}$; therefore (2.1) can be simplified to

$$f(t) = \iint_{S_+} \mu(\alpha, \beta) d\alpha \, d\beta - \iint_{S_-} \mu(\alpha, \beta) d\alpha \, d\beta. \tag{2.4}$$

From the above description, the output of Preisach model depends on the subdivision of triangle Ω. Equation (2.4) and its geometric representation is pure mathematical form of phenomenological Preisach model that was presented by Krasnoselskii. In order to implement the Preisach model for experimental applications, this primitive mathematical form of Preisach model was extended into a simple numerical form by Mayergoyz [25] that is discussed in the next section. It is worth mentioning that as an alternative method, if the form of density function in (2.4) (i.e., $\mu(\alpha, \beta)$) is predicted with respect to some experimental data, then the output of system can be easily calculated and this approach is described in Section 4.

3. Numerical Preisach Model

In addition to some numerical methods for Preisach density function approximation in hysteresis identification, there is a well-known and simpler numerical representation form of

Preisach model in which output of hysteresis system is calculated by summation of specific terms. These terms depend on history of local maximum and minimum of system input by specific functionality. This method is discussed precisely in this section.

At first, assume that input diagram in Figure 4 is applied to the target hysteresis system. It is worth mentioning that since, in most applications of SMA actuators, the input is electrical current and the current does not get negative value, in these cases the Preisach plane is in the first quarter of coordinate system. Let input increase from 0 to α_1 in time t_1, so all hysteresis operators with $\alpha_i < \alpha_1$ are on upper switching value and the others are on lower switching state. As it is discussed before, geometrically it results in subdivision of α-β plane triangle by line $u(t) = \alpha$ (see Figure 5). Upward motion through the line $u(t) = \alpha$ in Preisach plane is terminated when input reaches the local maximum value α_1. By using (2.4), output y_{α_1} in this time can be expressed as

$$y_{\alpha_1} = \iint_{S^+_{\alpha 1}} \mu(\alpha, \beta) \mathrm{d}\alpha \, \mathrm{d}\beta. \tag{3.1}$$

Then suppose that input is monotonically decreased to value β_1 in time t_2. As the input is being decreased through the line $u(t) = \beta$ in Preisach plane, all elemental operators in subdivision S^+ with $\beta_i > u(t)$ are turned off; so their outputs become zero. It makes triangle S^+ in Figure 5 divide into two regions (see Figure 6); therefore S^+ triangle region in Figure 5 has been switched to a trapezoid in Figure 6. This right-to-left motion of $u(t) = \beta$ line is terminated when input reaches local minimum value β_1. At this time, output $y_{\alpha_1\beta_1}$ can be expressed as

$$y_{\alpha_1\beta_1} = \iint_{S^+_{\alpha 1 \beta 1}} \mu(\alpha, \beta) \mathrm{d}\alpha \, \mathrm{d}\beta. \tag{3.2}$$

As a conclusion of this analysis, vertical and horizontal boundary lines between S^+ and S^0 regions in triangle α-β plane illustrate history of previous local maxima and minima of input, which affects current output value. This staircase boundary line is termed by memory interface $L(t)$. By generalizing the above discussion, at every instance of time, the output can be expressed as

$$y(t) = \iint_{S^+(t)} \mu(\alpha, \beta) d\alpha \; d\beta. \tag{3.3}$$

If function $F(\alpha, \beta)$ is defined as

$$F(\alpha, \beta) = y_\alpha - y_{\alpha\beta} \tag{3.4}$$

it is equal to the output increments along the first-order transition curves. These curves are defined as follows: the input of the system hysteresis is monotonically increased from zero to some value α and then decreased to a value β, that is, greater than zero and smaller than α. The term "first order" is used to emphasize the fact that each of these curves is formed after the first reversal of the input. It should be mentioned that the order of a minor loop branch is

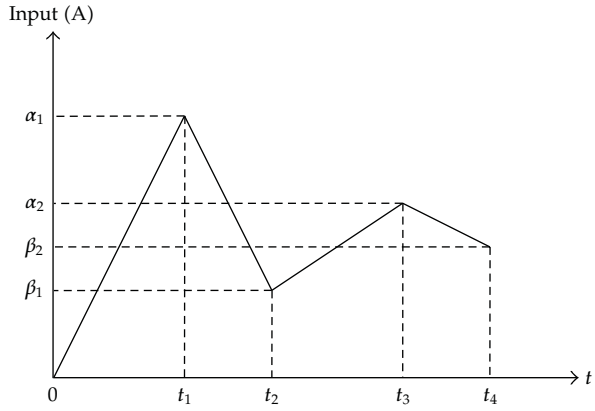


Figure 4: Input diagram.

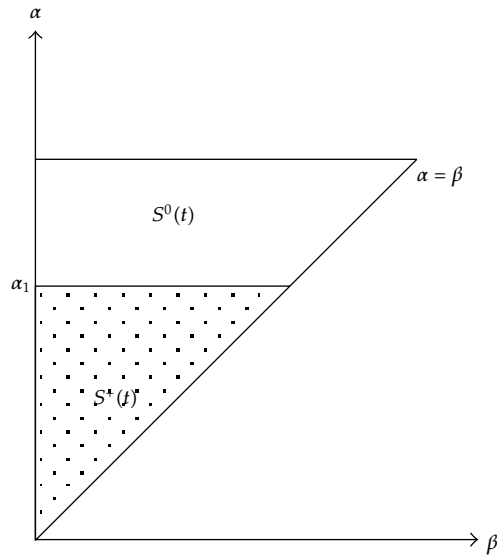


Figure 5: α-β plane due to input increasing to α_1.

defined to be the number of times that a hysteresis curve has reversed from a branch of the major hysteresis loop. Thus, all first-order branches are attached to one of the two branches of the major hysteresis loop. Similarly, the second-order branches are attached to the first-order branches, and the nth-order branches are attached to the $n-1$th-order branches [26].

From Figure 6 and (3.1), (3.2), (3.4),

$$\iint_{T(\alpha_1,\beta_1)} \mu(\alpha,\beta)\,d\alpha\,d\beta = F(\alpha_1,\beta_1), \tag{3.5}$$

where $T(\alpha_1,\beta_1)$ is a triangle which is shown in Figure 6.

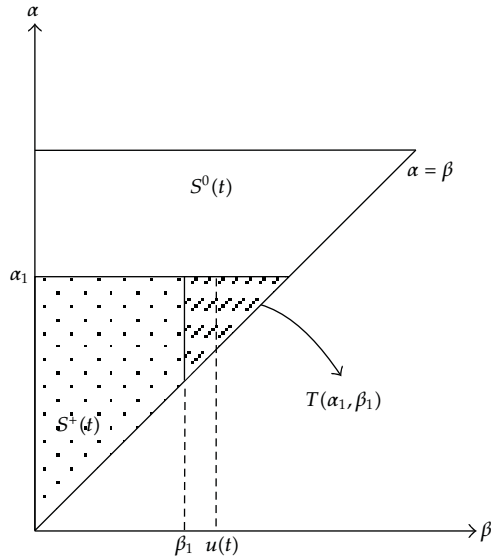


Figure 6: α-β plane due to input decreasing to β_1 from α_1.

In the case that the input is monotonically increasing, $S^+(t)$ can be represented as trapezoidal regions plus one triangle region ($Q_k(t)$ in Figures 7 and 8). Moreover, each trapezoidal region can be represented by two triangular region subtractions:

$$\iint_{Q_k(t)} \mu(\alpha,\beta)\,\mathrm{d}\alpha\,\mathrm{d}\beta = \iint_{T(M_k,m_{k-1})} \mu(\alpha,\beta)\,\mathrm{d}\alpha\,\mathrm{d}\beta - \iint_{T(M_k,m_k)} \mu(\alpha,\beta)\,\mathrm{d}\alpha\,\mathrm{d}\beta, \tag{3.6}$$

where M_k and m_k denote maximum and minimum of input history. From (3.5) it is known that

$$\begin{aligned} \iint_{T(M_k,m_k)} \mu(\alpha,\beta)\,\mathrm{d}\alpha\,\mathrm{d}\beta &= F(M_k,m_k), \\ \iint_{T(M_k,m_{k-1})} \mu(\alpha,\beta)\,\mathrm{d}\alpha\,\mathrm{d}\beta &= F(M_k,m_{k-1}). \end{aligned} \tag{3.7}$$

Using (3.6)–(13), it is concluded that

$$\iint_{Q_k(t)} \mu(\alpha,\beta)\,d\alpha d\beta = F(M_k,m_{k-1}) - F(M_k,m_k). \tag{3.8}$$

Thus, in the case of decreasing input, as shown in Figure 8, the final link of boundary interface line $L(t)$ is vertical line $m_n = u(t)$ and output can be calculated as

$$y(t) = \sum_{k=1}^{n(t)-1} [F(M_k,m_{k-1}) - F(M_k,m_k)] + F(M_n,m_{n-1}) - F(M_n,u(t)). \tag{3.9}$$

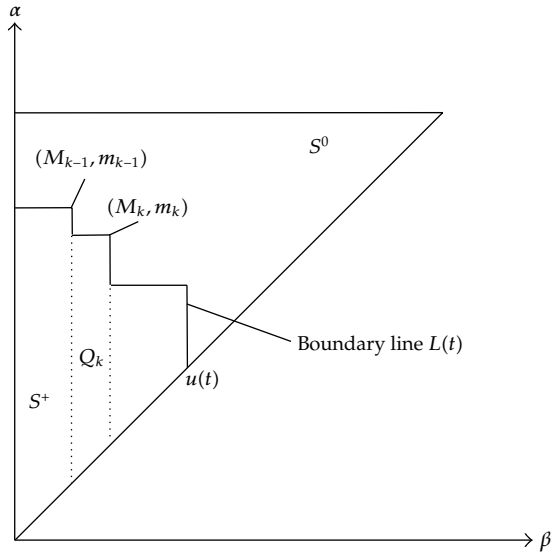


Figure 7: Numerical implementation of the Preisach model in the case of increasing input.

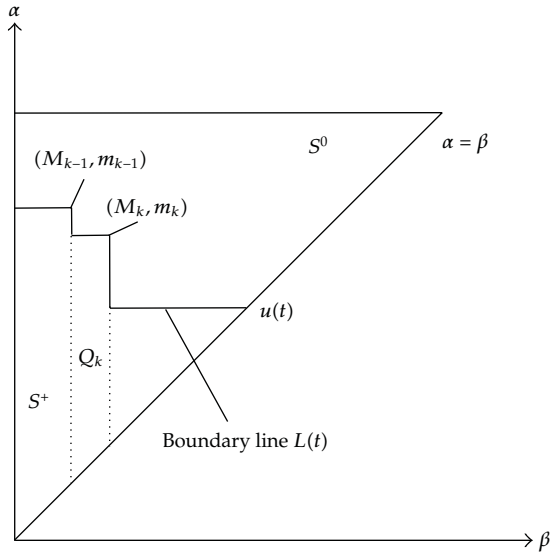


Figure 8: Numerical implementation of the Preisach model in the case of decreasing input.

Consequently, in the case of increasing input, as shown in Figure 7, the final link of boundary interface line $L(t)$ is horizontal line $M_n = u(t)$ and output can be determined as

$$y(t) = \sum_{k=1}^{n(t)-1} [F(M_k, m_{k-1}) - F(M_k, m_k)] + F(u(t), m_{n-1}). \tag{3.10}$$

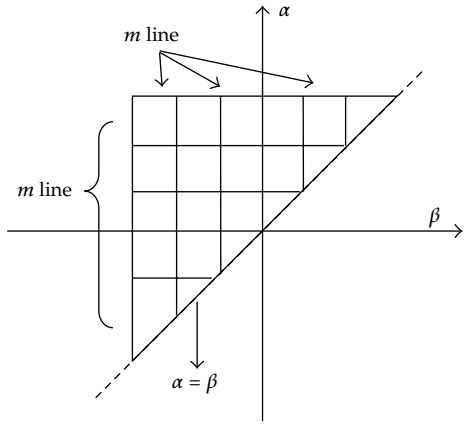


Figure 9: Discrete Preisach plane of Ω.

Therefore, it can be obviously seen that the output of numerical Preisach model can be calculated from current input value, as well as minima and maxima history terms. The $F(\alpha, \beta)$ function in desired points of α and β can be calculated by use of interpolation process on the batch of experimental data sample of hysteresis on the first-order reversal curves (ascending and descending). There are typical interpolation algorithms which can be used for this process such as cubic spline, nearest neighbor, linear interpolation, and Newton method.

4. Numerical Preisach Implementation

In order to implement the discussed numerical Preisach model, at first some experimental data is needed. By using the experimental data of the major ascending (or descending) loop and its attached first-order decreasing (or increasing) curves, a square mesh covering the Preisach plane Ω is obtained (see Figure 9). Any point (β_j, α_i) on the Preisach plane Ω can be considered as a vertex on a memory interface $L(t)$ which is formed by increasing input from the negative saturation state to a value α_i and then decreasing input to a state as β_j. Different pairs of inputs (β_j, α_i) with $\alpha_i \geq \beta_j$ from different first-order reversal curves inside the major ascending curve divide the Preisach plane into small cells. In the preprocessing stage (training process) of the numerical Preisach model by dividing the Preisach plane Ω uniformly, using m horizontal lines and m vertical l lines respectively, a total number of N nodes on the Preisach plan Ω can be computed as

$$N = \frac{(m+3)(m+2)}{3}. \tag{4.1}$$

In order to improve the prediction accuracy of the Preisach model, the equidistributed point should be large, and this greatly increases the complexity and cost of training process.

At the processing stage by entering an arbitrary input history and current values of input, the alternating series of dominant input extrema $\{M_k, m_k\}$ is first determined and for each new instant of time is updated. Finally, at the postprocessing stage (validation process), since most vertices of the memory interface $L(t)$ of the input do not coincide with those

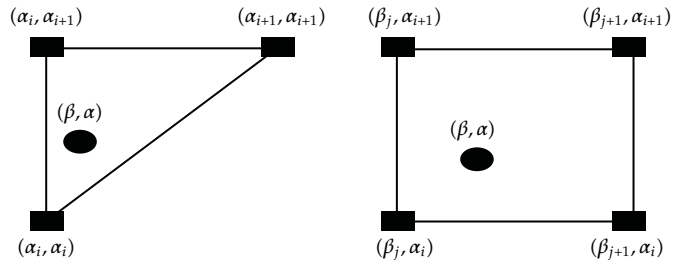


Figure 10: Calculation of $F(\alpha, \beta)$ in the postprocessing stage of numerical Preisach model by interpolation method.

discrete nodes of the discrete Preisach plane obtained in the preprocessing stage, the data of $F(\alpha, \beta)$ of these points should be obtained through interpolation [14].

This is done by first determining particular square or triangle cells to which points $(\mathrm{M_k}, \mathrm{m_{k-1}})$, $(\mathrm{M_k}, \mathrm{m_k})$, (M_n, m_{n-1}), $(M_n, u(t))$, and $(u(t), m_{n-1})$ belong. If a point (β, α) satisfies the condition $(\alpha - \alpha_i)(\alpha_i - \alpha_{i+1}) \leq 0$ and $(\beta - \beta_j)(\beta - \beta_{j+1}) \leq 0$, then this point is located inside a rectangular cell (see Figure 10) with vertices (β_j, α_i), (β_{j+1}, α_i), $(\beta_{j+1}, \alpha_{i+1})$, and (β_j, α_{i+1}) (for which the data of $F(\alpha, \beta)$ in these points are already experimentally measured in the processing stage). On the other hand if the point (β, α) satisfies the condition $(\alpha - \alpha_i)(\alpha - \alpha_{i+1}) \leq 0$ then the point (β, α) is located inside a triangle cell with vertices (α_i, α_i), $(\alpha_{i+1}, \alpha_{i+1})$ and (α_i, α_{i+1}). Next, the value of $F(\alpha, \beta)$ at those mentioned points can be computed by means of interpolation (cubic spline, nearest neighbor, linear interpolation, Newton method, etc.) of the mesh values of $F(\alpha, \beta)$ at the vertices of the above cells. Finally, the current values of output are evaluated by employing the formulae (3.9) and (3.10).

It is clear that in order to get the accurate results from the numerical Preisach model by the numerical interpolation methods, it is essential to have much data when the training process is performed. It means that the better Preisach plane Ω is divided in the training process (the larger m in (4.1)) and the results are more accurate for calculation of $F(\alpha, \beta)$ from (3.9) and (3.10) through the interpolation in the validation process.

5. Density Function Approximation

In this section a numerical density function approximation method proposed by Shirley and Venkataraman [16] is presented. It is based on mapping Preisach model into a linear equation system and tries to solve this equation in order to find best fit solution.

Often because of inadequate number of experimental data, there are some limitations to predict $\mu(\alpha, \beta)$. Let us consider α-β plane as a partitioned plane; so (2.4) can be restated as

$$f(t) = \sum_{i \epsilon S_{+1}} \mu_i(\alpha, \beta) + \sum_{j \epsilon S_{-1}} \left(-\mu_j(\alpha, \beta)\right). \tag{5.1}$$

In (5.1) it is supposed that a discrete density function with finite number of $\mu(\alpha, \beta)$ can be expressed by a vector and is denoted by $\psi(\alpha, \beta)$. Now, a state vector $(\widehat{S}_i)$ is defined,

indicating state of each $\gamma_{\alpha,\beta}$ (−1 or +1), in accordance with each input-output experimental data; thus, (5.1) can be formulized as

$$\begin{aligned} f_1 &= \widehat{S}_1(\alpha,\beta)\cdot\psi(\alpha,\beta),\\ f_2 &= \widehat{S}_2(\alpha,\beta)\cdot\psi(\alpha,\beta),\\ &\vdots\\ f_m &= \widehat{S}_m(\alpha,\beta)\cdot\psi(\alpha,\beta). \end{aligned} \tag{5.2}$$

Let $F = \begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix}$ and $S = \begin{bmatrix} \widehat{S}_1 \\ \vdots \\ \widehat{S}_m \end{bmatrix}$; so (5.1) can be modeled as a linear equation problem with unknown variable ψ as

$$F = S \times \psi(\alpha,\beta). \tag{5.3}$$

For m input experimental data, it is necessary to find the best fit density function which satisfies desired output. When m would be large enough, this linear equation system has no single solution. Therefore, the following equation should be minimized in order to reach this purpose:

$$\varepsilon = \|S\psi - F\|^2 \quad \psi > 0, \tag{5.4}$$

where ψ is target vector variable with n element same as the number of finite $\gamma_{\alpha_i\beta_j}$. It should be mentioned that one limitation happens when m is not comparable with n (i.e., $\operatorname{rank}(S) < \min\{m,n\}$). To overcome this problem, let $\operatorname{rank}(S) = q < \min(m,n)$. Then, by performing the singular decomposition on S^TS, there is $S^TS = QAQ^T$, where A is an $n \times n$ diagonal matrix and $\operatorname{rank}(A) = q < n$. After zero rows and zero columns elimination of matrix A as well as removing the corresponding columns of Q, we have $\tilde{A}$ and $\tilde{Q}$ for which $\tilde{Q}\tilde{A}\tilde{Q}^T = S^TS$; thus (5.3) and (5.4) can be modified as the following equation which should be minimized in order to find Z and then ψ:

$$\varepsilon = Z^T\tilde{A}Z - F^TS\tilde{Q}Z \quad \psi = \tilde{Q}Z. \tag{5.5}$$

In the sequel, the density function can be approximately calculated for a set of experimental data. The main advantage of Shirley method is its good ability of hysteresis systems identification without definition any of pre-defined density function. But, it is still not convenient enough in control applications especially in real-time control. It comes back to the fact that solving the above optimization problem is so time-consuming process.

6. Preisach Neural Network Approach

Regarding (3.9) and (3.10), it is illustrated that in numerical Preisach modeling, output value depends on local minima and local maxima, of input history by $F(\alpha,\beta)$ functionality.

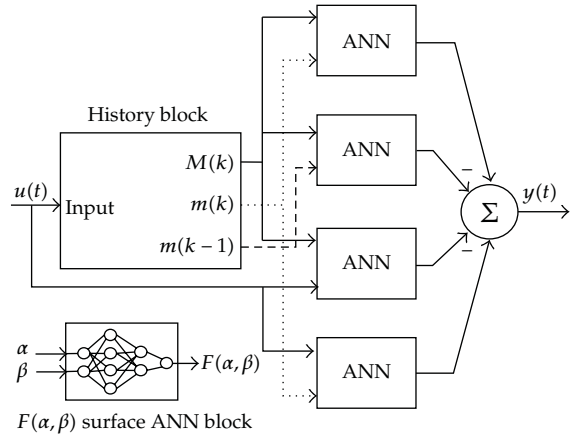


Figure 11: The proposed ANN-based Preisach model.

Consequently, this model needs a memory block which saves M_k, local maxima, and m_k, local minima of input at previous instants of time. $F(\alpha, \beta)$ is a suppositional surface on α-β plane which depends on hysteresis nature and can be defined numerically by experimental data of real system [25].

ANN is a computing system made up of a number of simple, highly interconnected processing elements, neurons, which processes information in parallel by its dynamic state response to external inputs [27]. Also ANN has powerful fault tolerant computing ability which has been used to model a wide range of systems for which mathematical models either cannot be defined or are ill-defined. Similarly, ANNs are well suited for systems that involve complex, multivariable processes with time variant parameters. As a result, ANN seems to be a good and powerful tool for approximation of $F(\alpha, \beta)$ surface.

In Figure 11, the proposed ANN-based numerical Preisach model is shown. In this method surface of $F(\alpha, \beta)$ is realized by two-dimensional input and single output feed-forward multilayer Perceptron neural network structure. It has two hidden layers with 15 and 5 neurons with tangent sigmoid activation function, while the activation function of output layer neurons is linear. Also, learning algorithm is Back-Propagate (BP). BP is based on error correction learning rule and can be considered as an extension of the mean least squares algorithm. This learning algorithm is used to adjust the network weights and biases in order to minimize the output error of the network [28]. The process of presenting batches of training cases to the network continues till the average error over the entire training set reaches a defined error goal or any other convergence criterion is achieved.

The history block works like a decision box and prepares M_k and m_k values for ANN base processing layer. In other words, it compares current input with the last input and updates maxima and minima values. It is considerable that history block should have wiping-out property. It means that each local maximum wipes out the vertices whose α coordinates are below this maximum, and each local minimum wipes out the vertices whose β coordinates are above this minimum. It is equivalent to the erasing of the history associated with these vertices. Thus, subsequent variations of input might erase some previous history [25].

In the next section, evaluation of presented approach is compared with the numerical classical Preisach method as well as the discussed Shirley method, precisely.

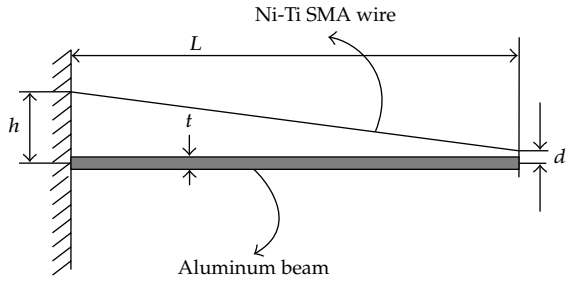


Figure 12: General structure of experimental apparatus.

Table 1: Parameters of experimental apparatus.

Parameter	l	t	d	E	b (beam width)	h
Value	400 mm	1.27 mm	100 mm	70 GPa	25 mm	100 mm

7. Experimental Results

Today there is wide range of SMA commercial applicability in many devices such as microrobots, medical and dental tools, nonexplosive aerospace actuators, and pneumatic microvalve [29]. In addition, recently SMA wires are being used as an intelligent material in new generation smart structures, which the shape of the structure would be controllable by use of SMA actuators. However, due to the deflection and displacement hysteresis behavior in SMA and Piezo actuators, there is rigorous hysteresis nonlinearity in these actuators which make their modeling and control more difficult. Consequently, accurate identification of these actuators behavior is one of the interesting and challenging fields in automation and control.

For evaluation of the proposed ANN-based Preisach hysteresis model, numerical classical Preisach model, and Shirley approach, a one-dimensional flexible aluminum beam whose deflection is controlled by an SMA wire as an actuator is used. This SMA actuator is made of Nitinol (Ni-Ti) alloy which has excellent electrical and mechanical properties, long fatigue life, and high corrosion resistance, and due to these properties this material is used in many SMA actuators today [30].

General structure of the experimental apparatus is shown in Figure 12. In this experimental setup a Flexinol TM actuator wire, manufactured by Dynalloy Inc, is used. This Ni-Ti SMA actuator wire is a one-way high-temperature (90°C) shape memory with 0.01-inch diameter. Parameters of the experimental apparatus and SMA wire according to Figure 12 are presented in Tables 1 and 2, respectively. The beam end deflection was measured by a high-precision linear potentiometer and fed to the computer through a multifunction (A/D, D/A) Advantech PCI-1711 card. The actuation input of the experimental apparatus is the current applied to the SMA wire and obtained from a D/A card and a V/I converter.

Since the input of the hysteresis model is assumed to be the temperature of the SMA wire, it is essential to determine the temperature of the wire. The temperature of the SMA wire is measured by two very thin (0.02 mm probe diameter) J thermocouples attached by a very conductive paste to both ends of the SMA wire. The average value obtained by the thermocouples is selected as the SMA wire temperature and fed to the computer through the mentioned multifunction card. This system is controlled in real time with real-time Windows Target Toolbox of MATLAB.

Table 2: Physical parameters of SMA wire actuator.

Parameter	Value
M_f	43.9°C
M_s	48.4°C
A_s	68°C
A_f	73.75°C
C_A	6.73 MPa/°C
C_M	6.32 MPa/°C
ε_L	4.10%
E_A	31.5 Gpa
E_M	20 Gpa
σ_s	25 MPa
σ_f	78 MPa

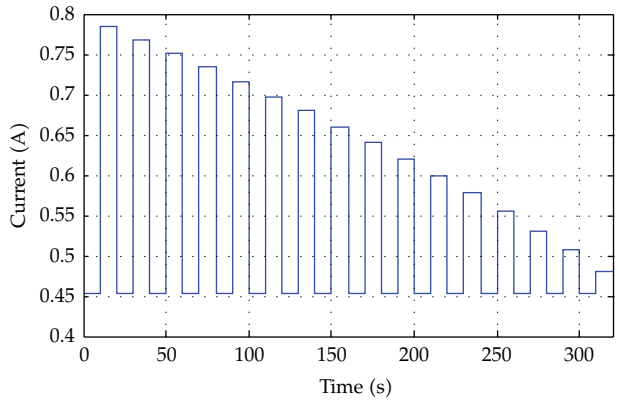


Figure 13: The decaying step current applied in the training process.

The input current applied to the SMA actuator in the training process is a decaying step signal and is shown in Figure 13. The corresponding beam deflection SMA wire temperature profile, obtained due to the applied current, is also shown in Figure 14. In the training process of the mentioned models 153 data set, consisting of the major loop and 15 first-order descending reversal curves, is used in order to approximate $F(\alpha, \beta)$ surface. For each switching point (α, β), according to (3.4), the corresponding $F(\alpha, \beta)$ is computed by measuring the output beam end deflection as the input temperature is increased to α and then decreased to β. Likewise, in Figure 15 surface of $F(\alpha, \beta)$ function which has been realized by ANN is also presented. Also to identify hysteresis system by using Shirley approach, density function should be approximated too. In the five sections, we described how this method can realize Preisach model by density function approximation based on finding best fit solution of a linear equation system. For hysteresis system modeling by Shirley approach, the same data set was used with 80 partitions for α-β plane [16]. The identified density function by this method is presented in Figure 16.

For evaluation of the prediction of the output beam end deflection by the numerical classical Preisach model, the proposed ANN-based Preisach model, and the Shirley approach, with respect to the experimental data, in the first validation process the current profile shown in Figure 17 is applied to the SMA actuator. In Figure 18 the prediction of the hysteresis

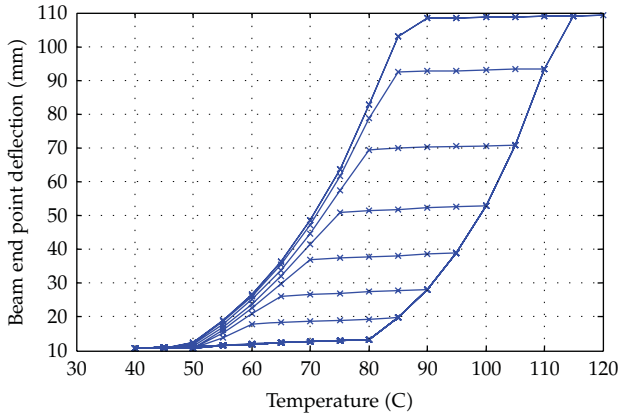


Figure 14: Hysteresis behavior between the beam end point deflection and the SMA wire temperature in the training process.

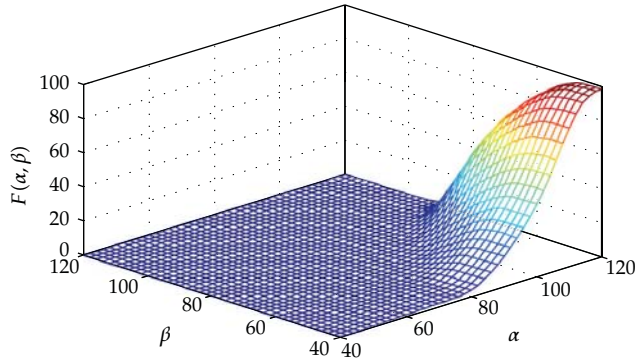


Figure 15: Surface $F(\alpha, \beta)$ identified by ANN using training data set.

behavior between the beam end point deflection and the SMA temperature by the Preisach model, the Shirley method as well as the proposed ANN method is compared by the experimental data. As it is seen from this figure none of the first-order transition curves applied in this validation process is applied in the training process. As it is clear in Figure 18 the proposed ANN based Preisach model can identify hysteresis in first order reversal curves more accurately than both the classical numerical Preisach model and the Shirley method. In order to show this property more clearly, the absolute error of the three mentioned models with respect to the experimental data is shown in Figure 19. The maximum, mean, and mean square values of absolute error for the three methods are also presented in Table 3.

For better evaluation of the three mentioned methods in predicting the hysteresis behavior of higher-order minor loops, in the second validation process the damped current profile shown in Figure 20 is applied to the SMA actuator. The prediction of the hysteresis behavior of higher-order minor loops by the Preisach model, Shirley approach, and the proposed ANN method is compared by the experimental data in Figure 21. This figure demonstrates the power ability of the proposed ANN model, with respect to two other models, in higher order reversal curves prediction. Indeed, it comes back to the general approximation capability of ANNs. Also, Figure 22 shows the absolute error of the three considered models with respect to the experimental data. In addition, the maximum, mean, and mean square values of absolute error for the three methods are presented in Table 4.

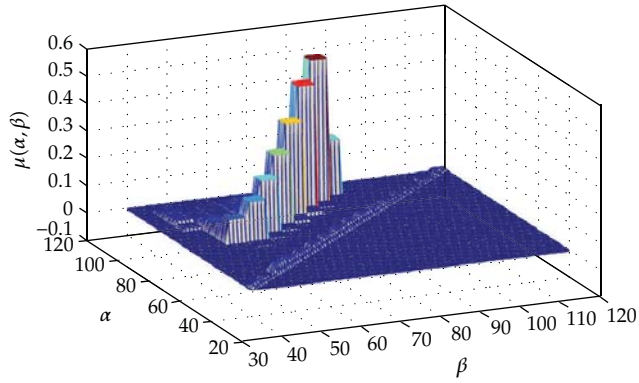


Figure 16: Approximated $\mu(\alpha, \beta)$ surface by use of linear equation system optimization (Shirley Approach).

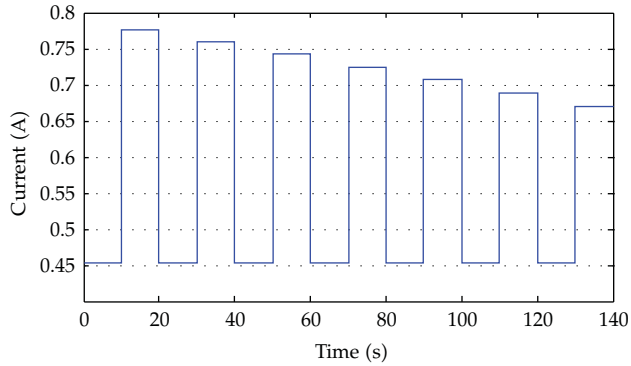


Figure 17: The decaying step current applied in the first validation process.

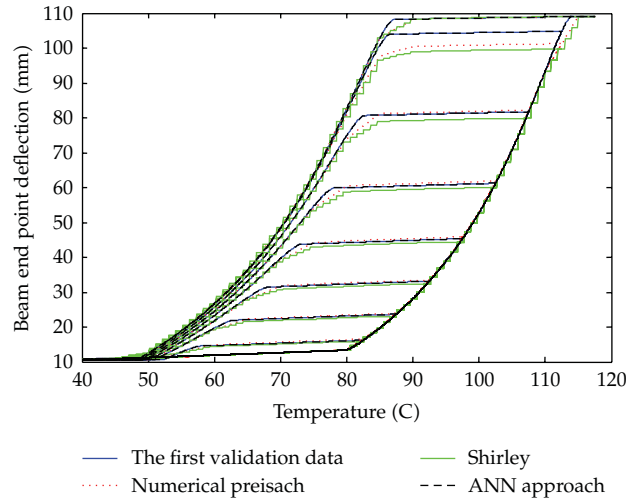


Figure 18: Prediction of the hysteresis behavior for the classical numerical Preisach, Proposed ANN Model and Shirley Approach in comparison to the first experimental data set.

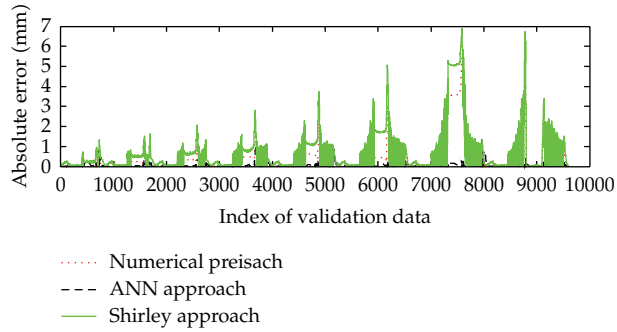


Figure 19: Absolute error of the Classical Numerical Preisach model, the Proposed ANN Model and the Shirley Method in comparison with the first experimental data set.

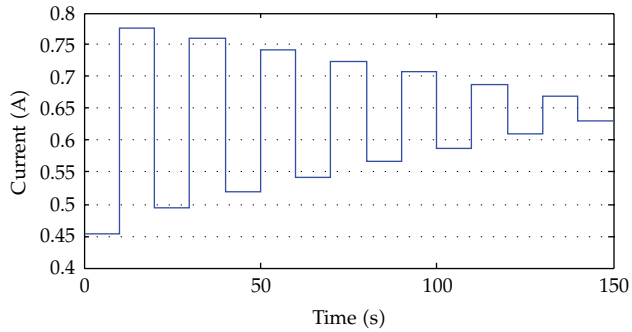


Figure 20: The decaying step current applied in the second validation process.

Table 3: Error of the three considered models in the first validation process.

	Mean absolute error (mm)	Max of absolute error (mm)	Mean square of absolute error (mm)
Numerical Preisach model	0.3228	5.4439	0.6145
Shirley method	0.644	6.907	1.505
Proposed ANN model	0.1315	1.36	0.25

Table 4: Error of the three considered models in the second validation process.

	Mean absolute error (mm)	Max of absolute error (mm)	Mean square of absolute error (mm)
Numerical Preisach model	1.5147	5.3964	4.2677
Shirley Method	1.78	6.827	5.629
Proposed ANN model	0.46	3.533	0.549

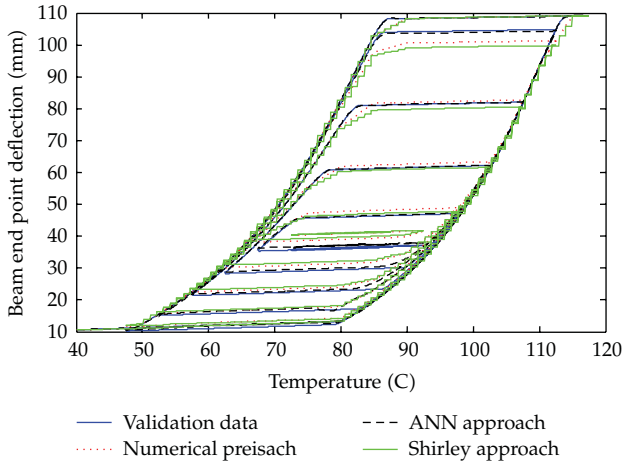


Figure 21: Prediction of the hysteresis behavior for classical numerical Preisach model, proposed ANN model, and Shirley approach in comparison to the second damped experimental data set.

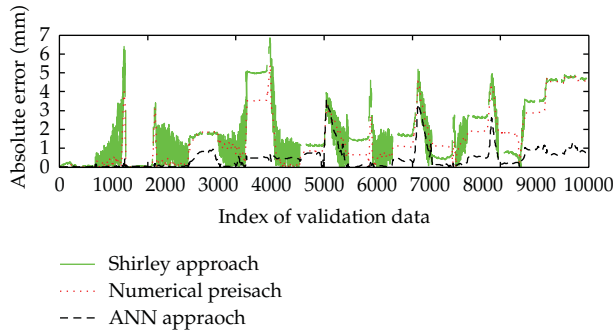


Figure 22: Absolute error of the classical numerical Preisach Model, the proposed ANN model, and the Shirley method in comparison with the second damped experimental data set.

As it is concluded from Figures 19 and 22 and Tables 3 and 4, the proposed ANN model has more accurate predictions than the numerical Preisach and Shirley models. In all of these validation processes the mean error of numerical Preisach model is better than the Shirley method while it is triple of corresponding error in the proposed ANN model. It is demonstrated (but not shown in this paper) that in order to bring the mean square error of the Preisach model (in the second validation process) at the order of the corresponding value of ANN model, the training should be done with 420 data instead of 153 data that was used first. It means that in order to have precise results in the numerical Preisach model, much data should necessarily be used, and this directly increases the experimental cost of training process. The significant decrease in the error of the proposed ANN model seems more valuable when it is known that in this ANN model, unlike the fuzzy inference engines, there is a straightforward training algorithm which enables utilizing this approach for many other hysteresis systems without any change in the structure of the ANN model.

8. Conclusion

In this paper, a novel hysteresis identification method based on numerical classical Preisach model by use of artificial neural networks (ANNs) has been presented. Since the accuracy of the Preisach function approximation methods is strongly dependent on the type of the candidate Preisach function and the number of its parameters, this approach remedies these drawbacks. In addition, this approach does not suffer from a lack of learning algorithms to adjust the system parameters existed in fuzzy inference engine.

The experimental data showed that this ANN model can predict SMA actuators hysteresis behavior with considerable accuracy in comparison with numerical Preisach model. It also has powerful ability to precisely predict the higher-order hysteresis minor loops behavior even though it is only trained by first-order reversal data. Therefore, it is a convenient method for many applications such as hysteresis nonlinearity control, hysteresis identification, and realization for performance evaluation in some physical systems such as magnetic and SMA materials.

References

[1] B. K. Nguyen and K. K. Ahn, "Feedforward control of shape memory alloy actuators using fuzzy-based inverse Preisach model," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 2, pp. 434–441, 2009.

[2] X. Tan and R. V. Iyer, "Modeling and control of hysteresis," *IEEE Control Systems Magazine*, vol. 29, no. 1, pp. 26–28, 2009.

[3] V. Basso, C. P. Sasso, and M. LoBue, "Thermodynamic aspects of first-order phase transformations with hysteresis in magnetic materials," *Journal of Magnetism and Magnetic Materials*, vol. 316, no. 2, pp. 262–268, 2007.

[4] S. Cao, B. Wang, R. Yan, W. Huang, and Q. Yang, "Optimization of hysteresis parameters for the Jiles-Atherton model using a genetic algorithm," *IEEE Transactions on Applied Superconductivity*, vol. 14, no. 2, pp. 1157–1160, 2004.

[5] J. V. Leite, S. L. Avila, N. J. Batistela et al., "Real coded genetic algorithm for Jiles-Atherton model parameters identification," *IEEE Transactions on Magnetics*, vol. 40, no. 2, pp. 888–891, 2004.

[6] K. K. Ahn and N. B. Kha, "Modeling and control of shape memory alloy actutors using Presiach model, genetic algorithm and fuzzy logic," *Journal of Mechanical Science and Technology*, vol. 20, no. 5, pp. 634–642, 2008.

[7] R. B. Gorbet, *Control of Hysteresis Systems with Preisach Represtations*, Ph.D. thesis, University of Waterloo, Ontario, Canada, 1997.

[8] S. Mittal and C. H. Menq, "Hysteresis compensation in electromagnetic actuators through Preisach model inversion," *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 4, pp. 394–409, 2000.

[9] X. Tan and J. S. Baras, "Modeling and control of hysteresis in magnetostrictive actuators," *Automatica*, vol. 40, no. 9, pp. 1469–1480, 2004.

[10] D. Hughes and J. T. Wen, "Preisach modeling of piezoceramic and shape memory alloy hysteresis," *Smart Materials and Structures*, vol. 6, no. 3, pp. 287–300, 1997.

[11] S. R. Viswamurthy and R. Ganguli, "Modeling and compensation of piezoceramic actuator hysteresis for helicopter vibration control," *Sensors and Actuators, A: Physical*, vol. 135, no. 2, pp. 801–810, 2007.

[12] K. K. Ahn and N. B. Kha, "Improvement of the performance of hysteresis compensation in SMA actuators by using inverse Preisach model in closed—loop control system," *Journal of Mechanical Science and Technology*, vol. 20, no. 5, pp. 634–642, 2006.

[13] K. K. Ahn and N. B. Kha, "Internal model control for shape memory alloy actuators using fuzzy based Preisach model," *Sensors and Actuators, A: Physical*, vol. 136, no. 2, pp. 730–741, 2007.

[14] I. D. Mayergoyz, *Mathematical Models of Hysteresis and their Applications*, Elsevier Science, New York, NY, USA, 2003.

[15] A. A. Adly and S. K. Abd-El-Hafiz, "Using neural networks in the identification of preisach-type hysteresis models," *IEEE Transactions on Magnetics*, vol. 34, no. 3, pp. 629–635, 1998.

[16] M. E. Shirley and R. Venkataraman, "On the Identification of Preisach Measures," in *Smart Structures and Materials 2003 Modeling, Signal Processing, and Control*, vol. 5049 of *Proceedings of SPIE*, pp. 326–336, September 2003.

[17] M. Cirrincione, R. Miceli, G. R. Galluzzo, and M. Trapanese, "A novel neural approach to the determination of the distribution function in magnetic Preisach systems," *IEEE Transactions on Magnetics*, vol. 40, no. 4, pp. 2131–2133, 2004.

[18] C. Natale, F. Velardi, and C. Visone, "Identification and compensation of Preisach hysteresis models for magnetostrictive actuators," *Physica B*, vol. 306, no. 1–4, pp. 161–165, 2001.

[19] E. Dlala and A. Arkkio, "A neuro-fuzzy-based Preisach approach on hysteresis modeling," *Physica B*, vol. 372, no. 1-2, pp. 49–52, 2006.

[20] J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[21] J. DA. Wei and C. T. Sun, "Constructing hysteretic memory in neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, no. 4, pp. 601–609, 2000.

[22] E. Kolman and M. Margaliot, *Knowledge-Based Neurocomputing*, Springer, Berlin, Germany, 2009, STUDFUZZ 234.

[23] F. Preisach, "Über die magnetische Nachwirkung," *Zeitschrift für Physik*, vol. 94, no. 5-6, pp. 277–302, 1935.

[24] M. A. Krasnoselskii and A. V. Pokrovskii, *Systems with Hysteresis*, Springer, Berlin, Germany, 1983.

[25] I. D. Mayergoyz, "Mathematical models of hysteresis," *IEEE Transactions on Magnetics*, vol. 22, no. 5, pp. 603–608, 1986.

[26] Z. Bo and D. C. Lagoudas, "Thermomechanical modeling of polycrystalline SMAs under cyclic loading, Part IV: modeling of minor hysteresis loops," *International Journal of Engineering Science*, vol. 37, no. 9, pp. 1205–1249, 1999.

[27] R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley, Reading, Mass, USA, 1989.

[28] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation, Parallel Data Processing*, vol. 1, MIT Press, Cambridge, Mass, USA, 1986.

[29] C. M. Wayman, T. W. Duerig et al., *Engineering Aspects of Shape Memory Alloys*, Butterworth-Heinemann, Oxford, UK, 1990.

[30] M. Novotny and J. Kilpi, "Shape Memory Alloys (SMA)," http://www.ac.tut.fi/aci/courses/ACI-51106/pdf/SMA/SMA-introduction.pdf.

Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2011, Article ID 414258, 14 pages
doi:10.1155/2011/414258

Research Article

An ANFIS Approach for Real Power Transfer Allocation

Hussain Shareef,[1] Saifulnizam Abd Khalid,[2] Mohd Wazir Mustafa,[2] and Azhar Khairuddin[2]

[1] *Faculty of Engineering and Built Environment, The National University of Malaysia, 43600 Bangi, Malaysia*
[2] *Faculty of Electrical Engineering, Technological University of Malaysia, 81310 Johor, Malaysia*

Correspondence should be addressed to Saifulnizam Abd Khalid, nizam@fke.utm.my

Received 17 June 2010; Accepted 13 September 2010

Academic Editor: Gani Stamov

This paper proposes an adaptive neurofuzzy interface system (ANFIS) approach to identify the real power transfer between generators. Based on solved load flow results, it first uses modified nodal equation method (MNE) to determine real power contribution from each generator to loads. Then the results of MNE method and load flow information are utilized to train the designed ANFIS. It also incorporated an enhanced feature extraction method called principle component analysis (PCA) to reduce the input features to the ANFIS. The 25-bus equivalent system of south Malaysia is utilized as a test system to illustrate the effectiveness of the ANFIS output compared to that of the MNE method. The ANFIS output provides promising results in terms of accuracy and computation time. Furthermore, it can be concluded that the ANFIS with enhanced feature extraction method reduces the time taken to train the ANFIS without affecting the accuracy of the results.

1. Introduction

The introduction of electricity privatization becomes an important issue under electric industry restructuring. The aim of this research is to bring transparency and open access to the transmission network. Implementing transparent rules that allocate transmission use fulfill this concept of fairness in the industry. Fairness can only be achieved by adopting a fair and transparent usage allocation methodology acceptable to all parties. In view of market operation, it is vital to know the role of individual generators and loads to transmission wires and power transfer between individual generators to loads. This is necessary for the restructured power system to operate economically, efficiently

and ensure open access to all system users [1]. Several schemes have been developed to solve the allocation problem in the last few years. Methods based on the Y-bus or Z-bus system matrices have recently received great attention since these methods can integrate the network characteristics and circuit theories into line usage and loss allocation. The method reported in [2] is based on Kirchhoff's current law (KCL), equivalent linear circuit that reaches all lines and loads. Based on the stated assumptions, a recursive procedure was used to construct the equivalent circuit for each bus. Moreover, superposition theorem was applied to the bus's equivalent circuit starting from a bus whose injected currents were known. Another circuit concept method was proposed by Chang and Lu [3]. It was based on the system Y-bus and Z-bus matrix modification where, branch current are determined as a function of generators' injected current by using information from the Z-bus. Similarly, contribution to bus voltages was computed as a function of each generator current injection by decomposing the network into different networks. Using the computed voltages and currents, the power flowing on the transmission lines were unbundled. It uses approximate formulation to calculate the unbundled loss components. This algorithm utilizes the network decomposition concept as proposed by Zobian and Ilić [4] which determines the use of transmission network by individual bilateral contracts. Teng [5], proposed a systematic method, very similar to that presented in [3], to allocate the power flow and loss for deregulated transmission systems. Using a similar concept, the authors of this paper introduce a modified nodal equation (MNE) method for real and reactive power allocation [6] in which the load buses powers are represented as a function of the generators' current and voltage.

The tracing methods [1, 7–10] based on the actual power flows in the network and the proportional sharing principles were effectively used in transmission usage allocation. The methods reported in [1, 9] are based on tracing the current and complex power from individual power sources to system loads. Based on solved load flow, the method converts power injections and line flows into real and imaginary current injections and current flows. This method has a clear physical meaning and its results are unique. Bialek in [7] proposed a novel power tracing method. However this method requires inverting a large matrix. Wu et al. [8] proposed graph theory to calculate the contribution factor of individual generators to line flows and loads and the extraction factor of individual loads from line flows and generators, which is theoretically efficient. This method cannot handle loop flows and losses must be removed initially. Reference [11] was based on the concept of generator "domains", "common", and "links". The disadvantage of this method is that the share of each generator in each "common" (i.e., the set of buses supplied from the same set of generators) is assumed to be same. Furthermore, the "commons" concept can lead to problems since the topology of a "common" could radically change even in the case of slight change in power flows. In a related work, a support vector machine (SVM) [12] was applied to estimate the contribution of individual generators to loads in power systems. The SVM gives faster results but the accuracy of the result is not promising.

Therefore in order to obtain fast and accurate results, an adaptive neurofuzzy interface system (ANFIS) approach is proposed in this work. The proposed method considers almost all system variables obtained from load flow solutions as the input features. However, these features are further reduced using the principle component analysis (PCA), to decrease the training time required by the designed ANFIS. The targets of the ANFIS corresponding to the real power transfer allocation results are obtained from MNE method. It is expected that the application of ANFIS can contribute in improving the computation time of real power allocation methodology for deregulated system.

2. Modified Nodal Equations Method

The derivation, to decompose the load real powers into components contributed by specific generators starts with basic equations of load flow. Applying Kirchhoff's law to each node of the power network leads to the equations, which can be written in a matrix form as [1]

$$I = YV, \tag{2.1}$$

where V is a vector of all node voltages in the system, I is a vector of all node currents in the system, and Y is the Y-bus admittance matrix.

The nodal admittance matrix of the typical power system is large and sparse, therefore it can be partitioned in a systematic way. Considering a system in which there are G generator nodes that participate in selling power and remaining $L = n - G$ nodes as loads, then it is possible to rewrite (2.1) into its matrix form as

$$\begin{bmatrix} I_G \\ I_L \end{bmatrix} = \begin{bmatrix} Y_{GG} & Y_{GL} \\ Y_{LG} & Y_{LL} \end{bmatrix} \begin{bmatrix} V_G \\ V_L \end{bmatrix}. \tag{2.2}$$

Solving (2.2) for I_L, the load currents can be presented as a function of generators' current and load voltages as

$$I_L = Y_{LG}Y_{GG}^{-1}I_G + \left(Y_{LL} - Y_{LG}Y_{GG}^{-1}Y_{GL}\right)V_L. \tag{2.3}$$

Then, the total real power P_L of all loads can be expressed as

$$P_L = \mathrm{Re}\{V_L I_L^*\}, \tag{2.4}$$

where $(*)$ means conjugate.

Substituting (2.3) into (2.4) and solving for P_L the relationship as shown in (2.5) can be found

$$\begin{aligned} P_L &= \mathrm{Re}\left\{V_L\left(Y_{LG}Y_{GG}^{-1}\right)^* I_G^* + V_L\left(\left(Y_{LL} - Y_{LG}Y_{GG}^{-1}Y_{GL}\right)V_L\right)^*\right\} \\ &= \mathrm{Re}\left\{V_L \sum_{i=1}^{nG} \Delta I_L^{*I_G} + V_L\left(\left(Y_{LL} - Y_{LG}Y_{GG}^{-1}Y_{GL}\right)V_L\right)^*\right\}, \end{aligned} \tag{2.5}$$

where

$$\left(Y_{LG}Y_{GG}^{-1}\right)^* I_G^* = \sum_{i=1}^{nG} \Delta I_L^{*I_G}, \tag{2.6}$$

where nG is the number of generators.

Now, in order to decompose the load voltage dependent term further in (2.5), into components of generator dependent terms, (2.8) is used. A possible way to deduce load node

voltages as a function of generator bus voltages is to apply superposition theorem. However, it requires replacing all load bus current injections into equivalent admittances in the circuit. Using a readily available load flow result, the equivalent shunt admittance Y_{Lj} of load node j can be calculated using

$$Y_{Lj} = \frac{1}{V_{Lj}} \left(\frac{S_{Lj}}{V_{Lj}} \right)^{*}, \tag{2.7}$$

where S_{Lj} is the load apparent power on node j and V_{Lj} is the load bus voltage on node j. After adding these equivalences to the diagonal entries of Y-bus matrix, (2.1) can be rewritten as

$$V = Y^{'-1} I_G, \tag{2.8}$$

where Y' is the modified Y of (2.1).

Next, adopting (2.8) and taking into account each generator one by one, the load bus voltages contributed by all generators can be expressed as

$$V_L = \sum_{i=1}^{nG} \Delta V_L^{*I_G}. \tag{2.9}$$

Now it requires a simple mathematical manipulation to obtain the required relationship as a function of generators dependent terms. By substituting (2.9) into (2.5), the decomposed load real powers can be expressed as

$$P_L = \mathrm{Re}\left\{ V_L \sum_{i=1}^{nG} \Delta I_L^{*I_G} + \sum_{i=1}^{nG} \Delta V_L^{*I_G} \left(\left(Y_{LL} - Y_{LG} Y_{GG}^{-1} Y_{GL} \right) V_L \right)^{*} \right\}. \tag{2.10}$$

This equation shows that the real power of each load bus consists of two terms by individual generators. The first term relates directly to the generators' current and the second term corresponds to their contribution to the load voltages. With further simplification of (2.10), the real power contribution that load j acquires from generator i is

$$P_{Lj} = \sum_{i=1}^{nG} P_{Lji}^{\Delta I_L} + \sum_{i=1}^{nG} P_{Lji}^{\Delta V_L}, \tag{2.11}$$

where $P_{Lji}^{\Delta I_L}$ is the current dependent term of generator i to P_{Lj} and $P_{Lji}^{\Delta V_L}$ is the voltage dependent term of generator i to P_{Lj}.

This allocation method has clear physical meaning as it takes into account the interaction between real and reactive power flows. Vector P_L is used as a target in the training process of the proposed ANFIS.

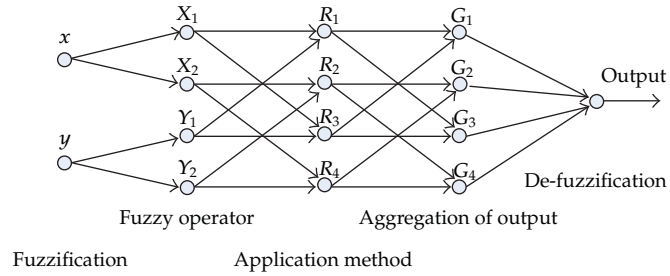


Figure 1: Basic structure of ANFIS.

3. Principle of Adaptive Neurofuzzy Inference System (ANFIS)

Adaptive Neurofuzzy Inference System (ANFIS) is developed from Sugeno-type fuzzy inference system (FIS) for effective data processing. The development is a simple data learning technique by using configuration of neurofuzzy model with hybrid learning rule. FIS processes a given input mapping to get a target output. The ANFIS defines five layers which perform the function of fuzzification of the input values, aggregation of membership degree, evaluation of the bases, normalization of the aggregated membership degree, and evaluation of function output values [13, 14].

Basic ANFIS structure consists the five main processing stages illustrated in Figure 1.

The first layer is the input layer which receives input data that are mapped into membership functions so as to determine the membership of a given input. In this fuzzification process the following equations are utilized:

$$X_i(x) = \frac{1}{\left[1 + \left((x - c_i)/a_i\right)^2\right]^{b_i}}, \quad i = 1, 2,$$
$$Y_i(y) = \frac{1}{\left[1 + \left((y - c_i)/a_i\right)^2\right]^{b_i}}, \quad i = 1, 2, \tag{3.1}$$

where X_i and Y_i are fuzzified input values, whereas a_i, b_i, and c_i are the parameter sets from the Gaussian input membership function.

The second layer of neurons represents association between input and output, by means of fuzzy rules. Application of fuzzy operators involves the use of the product (AND) to the fuzzified input. Equations (3.2) represent the fuzzy relations obtained from the product fuzzy operators:

$$R_1 = X_1(x) + Y_1(y),$$
$$R_2 = X_2(x) + Y_2(y),$$
$$R_3 = X_3(x) + Y_3(y),$$
$$R_4 = X_4(x) + Y_4(y). \tag{3.2}$$

In the third layer, the output is normalized and then passed to the fourth layer. Here, the activation degree and normalization is implemented by using the following equations

$$G_i = \frac{R_i}{(R_1 + R_2 + R_3 + R_4)}. \tag{3.3}$$

Then the output data are mapped in the fourth layer to give output membership function based on the predetermined fuzzy rules. Aggregation of all outputs is obtained by using (3.4) which is the product of the normalized activation degree and individual output membership function

$$O_i = G_i(p_i x + q_i y + r_i) \quad i = 1, 2, 3, 4, \tag{3.4}$$

where p_i, q_i, and r_i are the parameters from the output membership function.

Finally the outputs are summed in the fifth layer to give a single valued output. The ANFIS has the constraint that it only be designed as a single output system and the system must be of unity weights for each rule [15]

$$O = \sum_{i=1}^{4} O_i. \tag{3.5}$$

4. Feature Extraction

An important aspect to be considered for achieving good ANFIS performance is by proper selection and extraction of training data. For large-scale interconnected power systems, the complete state information is too large for any effective ANFIS implementation and therefore, the training data must be reduced to smaller number of useful information [16] using some sort of transformation. In general, the reduced set of features must represent the original set of features, since a loss of information in the reduced set results in loss of performance and accuracy of the ANFIS. The common methods for feature extraction are the linear discriminant analysis (LDA) and principle component analysis (PCA). In this work, PCA is used for feature extraction.

Obtaining the eigenvalues and eigenvectors of the covariance matrices for normal distributions is known as the principal component analysis (PCA). It is a statistical method often used in pattern recognition and data analysis. The goal of PCA is to map vectors having larger dimensional space onto another set of vectors having a smaller dimensional subspace. A brief explanation on calculation of principle components is as follows. Given a data set $X_{l\times m}$, where $l \in \Re^n$ represents the number of rows of the data X and $m \in \Re^n$ represents the number of input features of the data set. It is possible to calculate the mean value for the input features as

$$x_{\text{mean}} = \frac{(x_1 + \cdots + x_m)}{m} \tag{4.1}$$

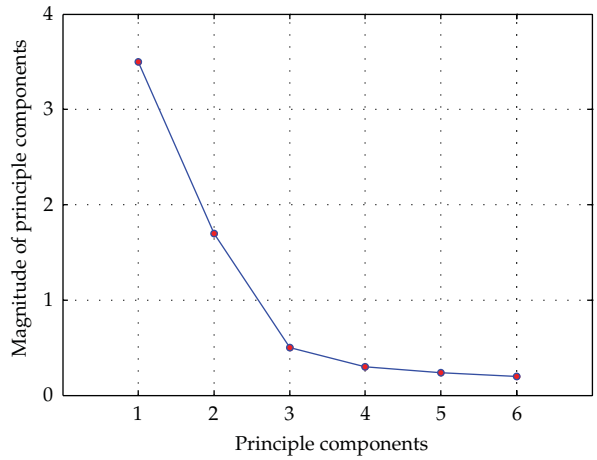


Figure 2: An example of a screen plot showing the principle components and its magnitude.

and subtracting the mean from the original features the covariance matrix can be obtained as

$$C = \frac{1}{l}\sum_{j=1}^{l}(x_1 - x_{\text{mean}}, \ldots, x_m - x_{\text{mean}})^T (x_1 - x_{\text{mean}}, \ldots, x_m - x_{\text{mean}}). \tag{4.2}$$

The final stage of CPA involves the eigenvectors and eigenvalues of the covariance matrix. The new obtained coordinates of the orthogonal projections onto the eigenvectors, are called principle components [16] where the number of principle components is equal to the number of input features. If too many principle components are considered, the transformed input features may include redundant features or if small number of principle components are chosen, they may jeopardize the accuracy of the intelligent system. One method of choosing principle components is by plotting them on a screen plot as shown in Figure 2 [17].

It can be noticed in Figure 2 that there is a "knee" in the plot at the third principle component; therefore according to a popular rule, the number of principle components to be considered should be 3 [17].

5. ANFIS Design for Real Power Allocation

In this work, 12 ANFIS blocks are generated and arranged as a hierarchical distribution ANFIS network to obtain real power transfer allocation results for the practical 25-bus equivalent power system of south Malaysia as shown in Figure 3.

This system consists of 12 generators located at buses 14 to 25, respectively. They deliver power to 5 loads, through 37 lines located at buses 1, 2, 4, 5, and 6, respectively. The data for training is assembled using the daily load curve and performing load flow analysis for every hour of load demand. Similarly the target vector for the training is obtained from the MNE method. Input data (D) for each developed ANFIS contains independent variables such as load bus voltage magnitude (V_{load}), real power of load (P_{load}), reactive power of load (Q_{load}), generator bus voltage magnitude (V_{gen}), real power generation (P_{gen}), reactive power generation (Q_{gen}) corresponding to that particular ANFIS block, average line real power

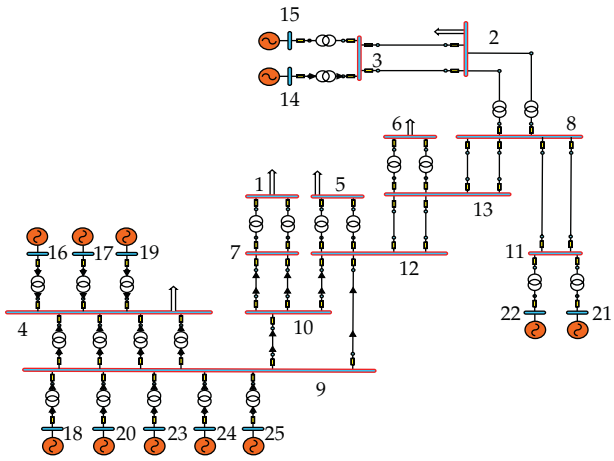


Figure 3: Single line diagrams for the 25-bus equivalent practical power system.

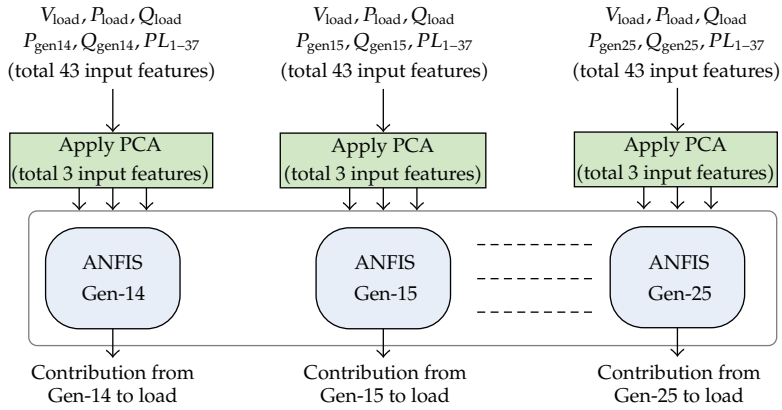


Figure 4: ANFIS design for real power transfer allocation for the 25-bus.

flows (P_{L1}, P_{L2}, to P_{L37}), and the target/output parameter (T) which is the contributions from a generator placed at particular bus to loads. This is considered as a single output from each ANFIS block for real power transfer allocation. Figure 4 shows complete ANFIS design for real power transfer allocation for the 25-bus equivalent practical power system assuming that system topology remains intact. Since PCA uses the eigenvectors of the covariance matrix and it only finds the independent axes of the data, the input data should not vary in abnormally wide range. Therefore for the effective use of PCA, it is assumed that the bus voltages, line flows, are power injections are within the limited range. This assumption is acceptable since the bus voltages, line power flows, generation limits generally varies in a narrow band to maintain voltage, and transient stability of the system. The observed maximum variation in the input samples for bus voltage, load power, generator power, and line power flow, respectively, are 0.037, 2.199, 0.380, and 0.535. Generally it is also noted that the variation in power flow and power injection increases as the load increase according to the daily load curve.

Table 1: Performance of individual ANFIS blocks.

ANFIS block	Mean error (%) ME(%)	Accuracy (%) (100-ME(%))	Prediction time (s)
Gen-14	0.0136	99.9864	0.003262
Gen-15	0.0136	99.9864	0.003287
Gen-16	0.0136	99.9864	0.003275
Gen-17	0.0048	99.9952	0.003249
Gen-18	0.0133	99.9867	0.003297
Gen-19	0.0216	99.9784	0.003396
Gen-20	0.0042	99.9958	0.003309
Gen-21	0.0261	99.9739	0.003207
Gen-22	0.0652	99.9348	0.003226
Gen-23	0.0018	99.9982	0.003247
Gen-24	0.0502	99.9498	0.00325
Gen-25	0.0432	99.9568	0.003367

5.1. Training

ANFIS is sensitive to the number of input features. Too many input features increasing training time. Therefore number of input features is selected by conducting PCA to eliminate those principle components that contribute less than 2% to the total variation in the original data set. After the PCA is applied, it is found that the total of input features can be reduced from 43 to only 3 input features without severely affecting the accuracy of the results.

After the reduced input features and target for training data is created, the data (D and T) is divided into training and test subsets. In this case 168 samples of data are used for the training and another 168 samples are created for testing. Figure 5 shows the performance of the training for individual ANFIS blocks representing each generator. From Figure 5, it can also be seen that the training goal is achieved in 3 epochs with a root mean square error less than 0.2×10^{-4}.

5.2. Pretesting and Simulation

After the ANFIS have been trained, next step is to simulate the ANFIS blocks. The entire test sample data is used in pretesting. After simulation, the obtained result from the trained blocks is evaluated with a linear regression analysis. The regression analysis for the trained ANFIS block that is referred to contribution of generator at bus 14 to loads is shown in Figure 6. The correlation coefficient, (R) in this case is equal to one which indicates a perfect correlation between MNE method and output of the ANFIS block. The best linear fit is indicated by the solid line, whereas the perfect fit is indicated by the dashed line.

6. Results and Analysis

A number of simulations have been carried out to demonstrate the accuracy of the developed ANFIS with the same 25-bus equivalent system of south Malaysia. The scenario here is a decrement by 5% of the real and reactive load demand from the nominal trained pattern. Besides it also assumed that all generators also decrease their production proportionally

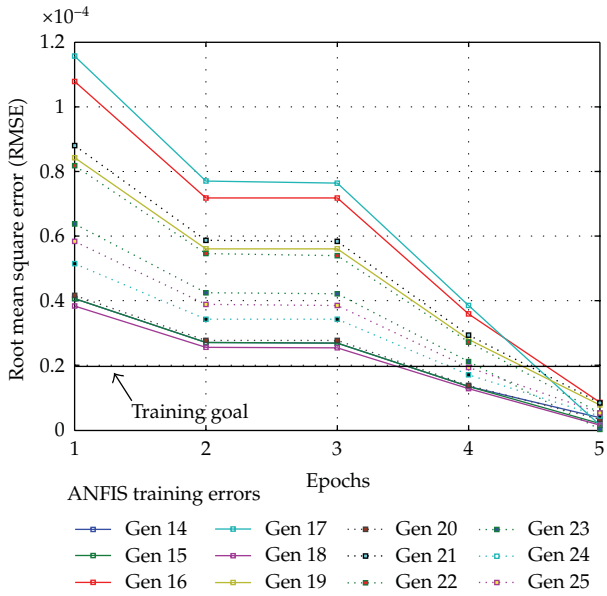


Figure 5: Training curves for individual ANFIS blocks.

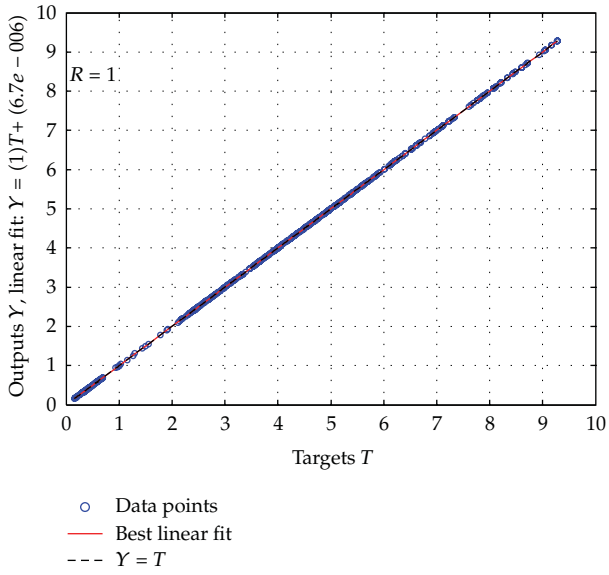


Figure 6: Regression analysis between the ANFIS output and the corresponding target.

according to this variation in the load demands. This assumption is being made to ensure that all real power generation of generator at buses 14 to 25 varies in responding the varying daily load pattern of the loads. Figure 7 shows the real power transfer allocation result for generator located at bus 14 calculated by the ANFIS along with the result obtained through MNE method for loads at buses 1, 2, 4, 5, and 6 for hours 25 to 48 and 121 and 144.

Table 2: Analysis of real power allocation for the 25-bus equivalent system during hour 33.

Supplied by	Load bus number									
(MW)	ANFIS Output					MNE Target				
	1	2	4	5	6	1	2	4	5	6
Gen-14	0.390	4.788	3.056	4.267	5.513	0.387	4.787	3.058	4.267	5.513
Gen-15	0.390	4.788	3.056	4.267	5.513	0.387	4.787	3.058	4.267	5.513
Gen-16	0.614	7.438	41.746	6.253	8.068	0.623	7.444	41.730	6.254	8.070
Gen-17	0.609	7.313	40.906	6.154	7.926	0.613	7.312	40.899	6.148	7.931
Gen-18	0.357	4.522	2.785	3.732	4.850	0.357	4.523	2.786	3.732	4.850
Gen-19	0.437	5.141	28.236	4.349	5.591	0.438	5.141	28.232	4.346	5.594
Gen-20	0.380	4.852	2.973	3.997	5.194	0.381	4.848	2.972	3.993	5.193
Gen-21	0.850	9.422	6.419	7.628	10.034	0.851	9.420	6.418	7.616	10.027
Gen-22	0.881	9.711	6.660	7.849	10.348	0.884	9.724	6.658	7.856	10.351
Gen-23	0.563	7.270	4.402	5.966	7.768	0.563	7.270	4.404	5.963	7.768
Gen-24	0.492	6.299	3.847	5.182	6.741	0.493	6.297	3.848	5.181	6.741
Gen-25	0.696	8.529	5.413	7.117	9.202	0.697	8.523	5.412	7.109	9.199
Total	6.660	80.072	149.499	66.759	86.748	6.673	80.076	149.476	66.730	86.749
Actual	6.673	80.076	149.476	66.730	86.749	6.673	80.076	149.476	66.730	86.749

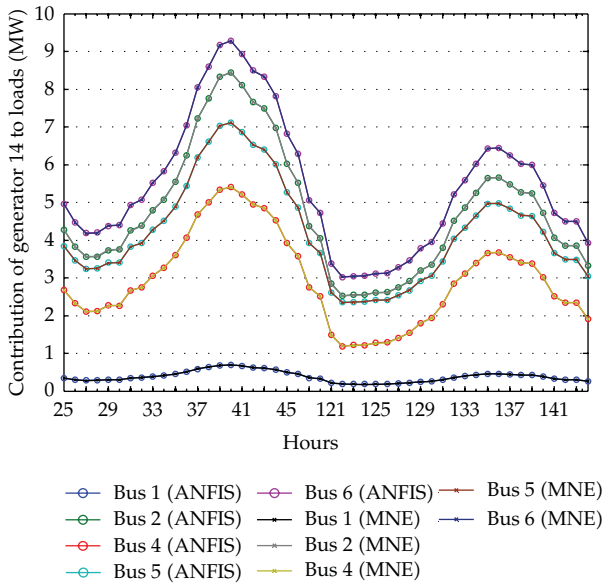


Figure 7: Distribution of real power from generator at bus 14 to loads within hours 25 to 48 and 121 and 144.

Results obtained from the ANFIS are indicated with lines having circles and the solid lines with cross represent the output of the MNE method. From Figure 7, it can be observed that the developed ANFIS can allocate real power transfer between generators and load with very good accuracy, almost 99.95%. In this simulation, ANFIS computes within 39.37 msec, whereas the MNE method took 360 msec for the calculation of same real power transfer allocation. Therefore it can be concluded that the ANFIS is more efficient in terms of computation time. Moreover it is worth highlighting that the feature reduction method used

Table 3: Load flow data for the 25-bus equivalent system at hour 33.

	Voltage		Generation		Load	
Bus no.	Magnitude (p.u)	Angle (p.u)	Real (MW)	Reactive (Mvar)	Real (MW)	Reactive (Mvar)
1	1.0442	8.3682	0	0	6.673	4.0038
2	1.0423	7.8052	0	0	80.076	24.023
3	1.0426	7.9571	0	0	0	0
4	1.0416	8.1925	0	0	149.48	56.053
5	1.031	6.1741	0	0	66.73	23.355
6	1.0335	5.6078	0	0	86.749	28.027
7	1.0455	8.4806	0	0	0	0
8	1.0559	8.2109	0	0	0	0
9	1.0462	8.626	0	0	0	0
10	1.0456	8.4878	0	0	0	0
11	1.0572	8.274	0	0	0	0
12	1.0447	8.2158	0	0	0	0
13	1.047	7.7774	0	0	0	0
14	1.04	9.4198	31.504	−2.6853	0	0
15	1.04	9.4198	31.504	−2.6853	0	0
16	1.05	9.88	36.987	10.629	0	0
17	1.05	9.8454	36.229	10.606	0	0
18	1.05	9.979	23.943	3.971	0	0
19	1.05	9.605	24.888	8.4126	0	0
20	1.05	10.086	25.833	4.0174	0	0
21	1.153	12.721	31.504	38.05	0	0
22	1.16	12.694	31.504	40.973	0	0
23	1.05	10.43	39.38	5.1706	0	0
24	1.05	10.17	33.709	5.0049	0	0
25	1.055	10.594	43.16	11.322	0	0

in this work is effective and it does not cause a noticeable error. Table 1 shows the percentage mean error, accuracy, and prediction time taken by individual ANFIS blocks. The percentage mean error, ME(%) is calculated from the following equations:

$$\mathrm{ME}(\%) = \sum_{n=1}^{N} \frac{E_n}{N} \times 100, \tag{6.1}$$

where E_n difference between desired output and actual output of test data number n and N is the total number of data.

The allocation of real power to loads using proposed ANFIS on hour 33 out of 168 hours is presented in Table 2 along with the result obtained through MNE method. From Table 2, it can be noted that the result obtained by the ANFIS output in this paper is compared well with the result of MNE method. The difference of real power between generators in both methods is very small which is less than or equal to 0.029 MW. The consumer located at bus 4 consumed the highest demand compared to other consumers in this hour. Consequently, the contribution of real power due to generators 16, 17, and 19 located at the same bus provides

more real power to load at bus 4 by both methods as well. This result also justifies the physical meaning of MNE method as these generators are nearest to load at bus 4. Moreover, to validate the results, load flow information at hour 33 for the test system are given in Table 3. It can be observed that the sum of the real power contributed by each generator obtained from MNE method and ANFIS is in conformity with the actual power flow.

7. Conclusion

In this paper, an ANFIS method has been developed to identify the real power transfer between generators and load. The developed ANFIS adopts real power allocation outputs determined by MNE technique as an estimator to train the ANFIS. The robustness of the proposed method has been demonstrated on the 25-bus equivalent system of south Malaysia. From the results, it can be concluded that the ANFIS output provides the results in a faster and convenient manner with good accuracy. Better computation time is crucial to improve online application. Hence, the proposed method could be adapted to true application of real power allocation and help to resolve some of the difficult real power pricing and costing issues to ensure fairness and transparency in the deregulated environment of power system operation.

Acknowledgments

The authors wish to acknowledge the Ministry of Higher Education, Malaysia (MOHE) for the financial funding of this project, and Universiti Kebangsaan Malaysia and Universiti Teknologi Malaysia for providing infrastructure and moral support for the research work.

References

[1] H. Shareef and M. W. Mustafa, "Real and reactive power allocation in a competitive market," *WSEAS Transactions on Power Systems*, vol. 1, pp. 1088–1094, 2006.
[2] R. Reta and A. Vargas, "Electricity tracing and loss allocation methods based on electric concepts," *IEE Proceedings: Generation, Transmission and Distribution*, vol. 148, no. 6, pp. 518–522, 2001.
[3] Y.-C. Chang and C.-N. Lu, "Electricity tracing method with application to power loss allocation," *International Journal of Electrical Power and Energy System*, vol. 23, no. 1, pp. 13–17, 2001.
[4] A. Zobian and M. D. Ilić, "Unbundling of transmission and ancillary services—part I: technical issues," *IEEE Transactions on Power Systems*, vol. 12, no. 2, pp. 539–548, 1997.
[5] J.-H. Teng, "Power flow and loss allocation for deregulated transmission systems," *International Journal of Electrical Power and Energy Systems*, vol. 27, no. 4, pp. 327–333, 2005.
[6] H. Shareef, M. W. Mustafa, S. Abd Khalid, A. Khairuddin, A. Kalam, and A. Maung Than Oo, "Real and reactive power transfer allocation utilizing modified Nodal equations," *International Journal of Emerging Electric Power Systems*, vol. 9, no. 6, pp. 1–14, 2008.
[7] J. Bialek, "Tracing the flow of electricity," vol. 143, pp. 313–320.
[8] F. F. Wu, Y. Ni, and P. Wei, "Power transfer allocation for open access using graph theory - fundamentals and applications in systems without loopflow," *IEEE Transactions on Power Systems*, vol. 15, no. 3, pp. 923–929, 2000.
[9] M. W. Mustafa, H. Shareef, and M. R. Ahmad, "An improved usage allocation method for deregulated transmission systems," in *7th International Power Engineering Conference, IPEC2005*, sgp, December 2005.
[10] S. Abdelkader, "Efficient computation algorithm for calculating load contributions to line flows and losses," *IEE Proceedings: Generation, Transmission and Distribution*, vol. 153, no. 4, pp. 391–398, 2006.
[11] D. K. Ron and A. G. Strbac, "Contributions of individual generators to loads and flows," *IEEE Transactions on Power Systems*, vol. 12, no. 1, pp. 52–60, 1997.

[12] H. Shareef, A. Mohamed, S. N. Khalid, M. W. Mustafa, and A. Khairuddin, "Real power transfer allocation utilizing support vector machine," in *Proceedings of The International Conference of Electrical Energy and Industrial Electronic Systems (EEIES '09)*, pp. 1–7, Penang, Malaysia, December 2009.

[13] P. Simon, *Oscillatory stability assessment of power system using computational intelligence*, Ph.D. thesis, Universit Duishdurg-Essen, Essen, Germany, 2005.

[14] J. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[15] S. M. Bateni, S. M. Borghei, and D.-S. Jeng, "Neural network and neuro-fuzzy assessments for scour depth around bridge piers," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 3, pp. 401–414, 2007.

[16] C. A. Jensen, M. A. El-Sharkawi, and R. J. Marks II, "Power system security assessment using neural networks: feature selection using fisher discrimination," *IEEE Transactions on Power Systems*, vol. 16, no. 4, pp. 757–763, 2001.

[17] B. Scholkopf, A. Smola, and K. R. Muller, *Kernel Principal Component Analysis, in Advances in Kernel Methods—SV Learning*, MIT Press, Cambridge, Mass, USA, 1999.

Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2011, Article ID 246286, 14 pages
doi:10.1155/2011/246286

Research Article

Downscaling Global Weather Forecast Outputs Using ANN for Flood Prediction

Nam Do Hoai, Keiko Udo, and Akira Mano

Disaster Control Research Center, Tohoku University, Aoba 6-6-11, Sendai 890-8579, Japan

Correspondence should be addressed to Nam Do Hoai, nam@potential1.civil.tohoku.ac.jp

Received 15 September 2010; Accepted 23 January 2011

Academic Editor: Gani Stamov

Downscaling global weather prediction model outputs to individual locations or local scales is a common practice for operational weather forecast in order to correct the model outputs at subgrid scales. This paper presents an empirical-statistical downscaling method for precipitation prediction which uses a feed-forward multilayer perceptron (MLP) neural network. The MLP architecture was optimized by considering physical bases that determine the circulation of atmospheric variables. Downscaled precipitation was then used as inputs to the super tank model (runoff model) for flood prediction. The case study was conducted for the Thu Bon River Basin, located in Central Vietnam. Study results showed that the precipitation predicted by MLP outperformed that directly obtained from model outputs or downscaled using multiple linear regression. Consequently, flood forecast based on the downscaled precipitation was very encouraging. It has demonstrated as a robust technology, simple to implement, reliable, and universal application for flood prediction through the combination of downscaling model and super tank model.

1. Introduction

Numerical weather prediction (NWP) has demonstrated its breakthrough in flood forecast recently. In terms of forecast lead time, the flood prediction based upon numerical weather prediction outputs tends to outperform other conventional forecasts which are based on real-time observation, especially in small- to medium-size basins where runoff concentration is relatively short. The forecast lead times given by NWP are ranging from short-term forecast (a couple of hours to few days) to medium-range forecast (up to ten days or more). This allows implementing effective action plans to minimize flood risk. At global scale, even though NWP models have showed significant improvement in terms of spatial resolutions, presently ranging from 25 to 50 km, these spatial resolutions are still far away

from the requirement for hydrological simulations or local-scale weather researches that usually require much finer resolutions, of about hundreds meters for small catchments to a couple of kilometers for large basins. Since land surface is averaged within very coarse grid cells; thus, small-scale effects of topography may not be resolved in the global NWP model. As a result, outputs from NWP models, hereinafter called the model outputs, are usually unreliable such as precipitation that is well known as the most unpredictable variable. For that reason, the flood forecast based on model outputs was not what had been expected [1, 2]. Although it is hoped that in the near future the global NWP models might be operational at finer resolutions, downscaling the model outputs from such resolutions to specific sites or area averages for practical uses is essential. Downscaling is a familiar technique used in climate research and weather forecast that aims to utilize information derived from the global NWP model outputs, particularly in the assessment of hydrological implication driven by global climate models or attempting for runoff prediction.

With respect to hydrological simulation and forecasting, though it is a new approach, artificial neural network (ANN) has been reported to give better performance in rainfall-runoff modeling than some other conventional runoff forecasts [3]. The use of (ANN) in runoff prediction is basically divided into two approaches. First, most studies used ANN approach for direct runoff or river stage prediction, as described in literature [4]. The prediction based on ANN is generally considered as an empirical method using mathematic transfer functions that relate stream flow with other weather variables. In this case, the rainfall-runoff process is known as a black box system whose inputs are usually hydrometeorological variables, such as antecedent rainfall and stream flow, and outputs are regularly the runoff or stage prediction. This process apparently omits the effect of physical characteristics of catchment on runoff generation processes. As a result, it sometimes reflects the inconsistency and inefficiency. In addition, this method usually provides the runoff forecast subsequence to the occurrence of rainfall. Therefore, the forecast lead-time is relatively short, especially the quick response catchments, of the order of zero to a couple of hours. To some extent, it is considered insufficient to put in place effective flood mitigation measures. On the contrary, the second approach employs ANN for indirect runoff prediction. The ANN is used to increase the accuracy of precipitation prediction, usually obtained from the model outputs, which is then used as inputs to hydrological models for runoff prediction. This approach is likely to outperform the previous one not only in terms of forecast lead-time extension but also the inclusion of the effects of catchment physical variations and rainfall distribution on the runoff generation process; thus, predictions are more realistic. Recent studies showed that potential predictors used for precipitation prediction are mostly based on outputs from high-resolution weather prediction models or combination of these outputs with other remote sensing information such as images of cloud structures [5, 6]. However, there have been existing limitations that these studies were restricted either just applicable for some in limited areas where are accessible to high-resolution models or at relatively short-term forecast. It means that greater lead times of flood forecast in larger scales are required, particularly in developing countries where globally covered NWP models are available; therefore, its benefit is maximized.

This paper presents the development of an empirical-statistical approach to downscale the precipitation from global NWP outputs to a basin-average scale for flood runoff prediction. The most popular ANN architecture, the feed-forward multilayer perceptron (MLP) using error training back-propagation method, was selected to downscale the large-scale precipitation. The large-scale predictors were obtained from the global NWP outputs, operated by Japan Meteorological Agency, with 0.5° grid point value data. Physical bases of

precipitation evolution were analyzed for the optimization of the MLP configuration used in calibration processes because it is very essential to drop out uncorrelated variables that cause overfitting and to overcome the requirement of a long historical record for learning stages as well as to speed up computational skills.

The study point of view targets to use simple approaches which are widely applied and of proven accuracy. Hence, the downscaled precipitation was then used as inputs to the super tank model, a calibration-free requirement rainfall-runoff model, which has been applied in simulating the river flow in a number of basins across various spatiotemporal scales to predict the flood runoff. The case study was conducted for the Upper Thu Bon River, located in Central Vietnam, where floods are considered the most dangerous calamity to human lives. The study results are expected to enhance the existing flood-forecasting technologies and mitigation practices. This is considered valuable for developing countries where ground weather observation is scarce and access to high resolution NWP models is limited.

2. Data and Methodology

2.1. Data and Study Area

Global NWP models are operational at many national meteorological agencies such as in Europe, America, and Japan. These models have been often upgraded; as a result, its spatiotemporal resolution has been quickly increased since the advent of computer technology. However, each agency has its own forecast purposes and is very much dependent on computational capability. The models might be different in terms of physical parameterization, forecast range, forecast issue routine, and spatiotemporal resolution. This study used atmospheric variables derived from the global NWP model outputs, operational at Japan Meteorological Agency, with spatial resolution of 0.5° and 60 vertical layers. It is currently considered the most advanced NWP model, for global scale, that produces 84-hour forecast, issued 4 times per day, at 00, 06, 12, and 18UTC. Precipitation on the surface and other variables are predicted for every 6-hour interval, 00–6, 6–12, 12–18, and 18–24UTC.

This study addressed a convention that precipitation obtained from rain gages was considered as reference rainfall (truth) for the comparison. Given the fact that the global NWP with 0.5° spatial resolution has been operational since late 2007, this analysis was based on archived data for the wet seasons, 2008 and 2009. Inverse distance weighting method was used to interpolate precipitation and related atmospheric parameters either from a point representation (rain-gage) or grid point value representation (NWP) to area-average basis.

In Central Vietnam, it has been highlighted that flood is the most common climate-related disaster. The region has often experienced large-scale floods during the wet seasons, from September to December, annually. These floods were usually caused by extremely widespread intense orographic rainfall that occurred on the windward of the Annam Range, known as the border between Vietnam and Laos. This type of rainfall was basically formulated through the combination of the cold surge from Northern continent and the tropical depression from the Pacific Ocean [7]. The basin selected in this study is the upper reach of Thu Bon River (see Figure 1) with the catchment area of 3,150 km^2. Average basin elevation is about 445 m above the mean sea level. Given the topographical features, rivers in this region are generally very short and steep; therefore, catchment response is rapid, resulted in very short time for people at risk to implement effective flood mitigation measures.

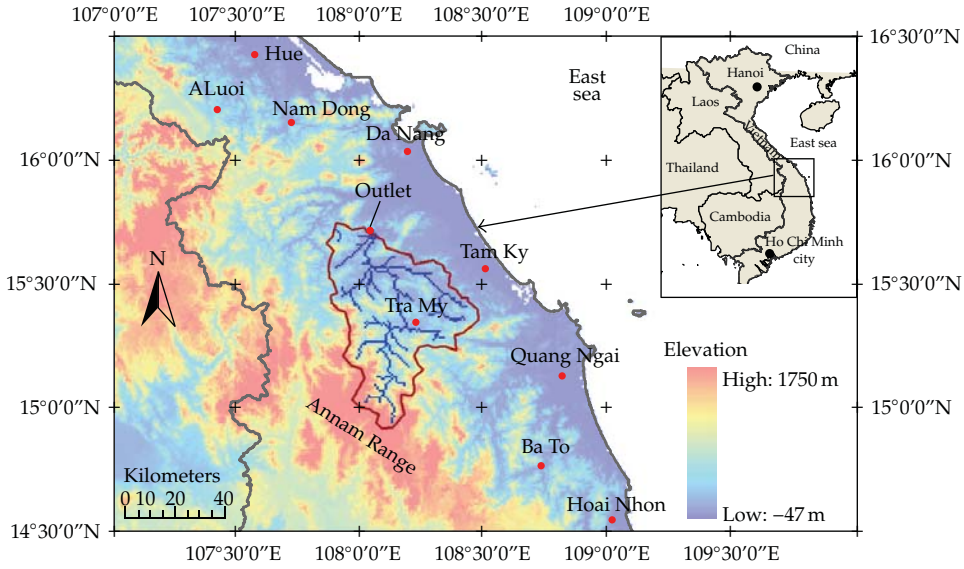


Figure 1: Location of study area and weather observation points (•).

2.2. *Downscaling*

It is expected that in the near future global NWP models may be operational at very fine resolution; however, there will still be a requirement to refine these models outputs to individual sites or local scales for weather research and application [8]. Downscaling is a common technique used in climate research and weather forecast that tries to utilize global atmospheric variables such as precipitation information for the use of impact assessment, hydrological simulation, and forecasting at subgrid or small scales. Downscaling methods are simply classified into dynamical approach and empirical-statistical approach [9].

The dynamical downscaling normally refers to limited area models (LAMs), or the so-called nested grid models, embedded with global NWP fields at its lateral boundaries. This type of models is usually based on fundamental physical principles and is able to take into account considerations of small-scale effects of land surface on weather patterns. Thus, outputs from LAM are expected to be realistic. However, it requires cost-effective evaluation regarding the selection of downscaling methods. First, it is costly to run LAM because of massive computational requirements, known as super computer systems. As a result, not many nations, especially the developing countries, are able to establish its own LAM for the purpose of weather forecast and hydrological simulations. This type of model is currently available only in such regions as North America (North Atlantic Model), Japan (Mesoscale Model), and some regions in Europe (COSMO Model for Germany). Second, the LAM are also subject to systematic biases as their accuracy is very much dependent on the veracity of the global weather model outputs that are used to derive the boundary conditions of the LAM [9]. In addition, in some cases finer-grid data might not be well reflected in the hydrological model if the details are aggregated over time and space [10].

In line with the scope of this study that simple approaches across a wide range of application are preferable the present study used statistical downscaling method that is considered as one of the most cost-effective methods in local-impact assessments of climate

scenarios and weather forecast. The statistical downscaling is cheap to run and universally applicable. Various statistical tools for weather downscaling have been proposed and were clearly reviewed by Wilbey and Wigley [8]. The statistical downscaling is fundamentally based on the formulation of either linear or nonlinear relationships between large-scale atmospheric variables and local or single-site scale variables. These relationships are then used to correct the outputs of the NWP models. The weakness of the method is that it requires a long historical weather observation record for the calibration processes. However, in the context of the present study, this problem can be avoided by taking into account the physical bases regarding the evolution processes of storm events in the study area. These storms are widespread orographic rainfall, typically following tropical depressions and typhoons in wet seasons.

Model output statistic (MOS) has been well known for a long history as a statistical downscaling tool for operational weather forecast [11]. Multiple linear regression is commonly used in the formulation of relationships between variables (predictors) derived from global NWP outputs and local or small-scale variables (predictants) such as downscaling precipitation. This method has been accepted in many meteorological centers for operational weather forecast, for example, in the USA. Nonetheless, precipitation is one of the most unpredictable variables, in terms of learning skill; a review of empirical downscaling techniques indicated that ANNs are generally observed to perform a better learning ability than the other regression-based downscaling techniques [4, 12]. Consequently, the present study employed ANN to downscale the large-scale precipitation derived from the global NWP model outputs.

Ground weather analysis based on downscaling global NWP outputs is typically conducted for individual points, subgrid scale or area-average bases. The NWP outputs are provided as a grid-point-value format, at very coarse spatial resolution. Using area-average downscaling approach apparently tends to reduce small-scale effects, in particular when the catchment size is larger than the size of grid cells (approximate 2,500 km^2). Information from neighbor grid points was preliminarily interpolated to the study basin using inverse distance weighting method. The same procedure was conducted for the ground rainfall observation points in order to compare with those obtained from the model outputs.

2.3. Artificial Neural Network (ANN)

Artificial neural network is simply understood as a nonlinear statistical data modeling tool that presents complex relationships between predictors (input layer) and predictants (output layer) through a synapse system (hidden layers) connecting predictors with predictants, or the so-called required outputs. As a result, ANN has demonstrated its wide range of application to solve complicated problems in many fields, for instance, engineering and environment.

Given many types of ANN have been extensively developed so far, as stated in literature [4], especially since the error back-propagation training algorithm was explored [13] it is very important regarding the selection of an appropriate ANN configuration and training method. In many cases, cost-effective analysis should be considered. Present study approach, for example, to employ a simple and reliable technology is preferable. One of the most simple and popular ANN architectures which was mostly used in hydrological modeling, approximately 89% [4], the feed-forward multilayer perceptron (MLP) using error back-propagation weight update rule, was employed for calibration processes. Presently,

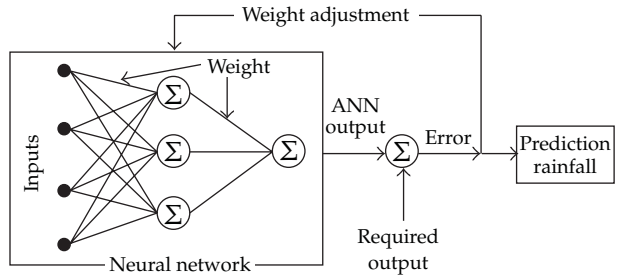


Figure 2: ANN architecture with back-propagation algorithm.

the MLP neural network was found to outperform radial basis function neural network and other multiple linear regression methods. The feed-forward MLP configuration selected here includes an input layer, a single hidden layer which has been selected by most researches [4], and an output layer that is interconnected by synapse weights (Figure 2). The number of nodes of the hidden layer was selected ranging from $(2n + 1)$ to $(2n^{0.5} + m)$, where n is the number of input nodes and m is the number of output nodes [14]. The training phase of the ANN is to adjust the weights so that the difference between the network outputs (predictants) and the expected outputs is minimized. For each node at a given layer, the outputs of n neurons in the previous layer provide the input to that node. These outputs are multiplied by the respective weights of connections between nodes, and then the summation function adds together all these products to produce the input that is processed by the activation function. The selection of activation functions is dependent on the type of network and learning algorithm; however, logistic sigmoid and hyperbolic tangent functions have been mostly employed, up to 64% and 13%, respectively [4]. Approximation used for the weight change is given in (2.1) by the delta rule [15]

$$w_{\text{new}} = w_{\text{old}} - \eta \frac{\partial E^2}{\partial w}, \tag{2.1}$$

where η is the learning rate parameter, w is the weights, and E^2 is the squared error.

2.4. Optimization of Predictors

It has been already addressed in [4], with respect to data handling for ANN, in which the determination of appropriate predictors for the input layer is very important. This process tends not only to drop out those variables that have less influence on the output to avoid overfitting but also to overcome the shortage of historical record used for calibration processes. For that reason, this study takes into account the physically based consideration regarding the precipitation evolution. As addressed in the previous sections, the study area is dominated by the orographic rainfall that intense rainfall is prevailing on the windward side (to the East) of the Annam Range of about 1500 m, that is, approximately equal to the geopotential height of 850 hPa pressure level, and very little precipitation falls on the leeward side (to the West). Therefore, among hundreds of variables provided by the global NWP outputs, only variables which are driving factors for the orographic rainfall evolution were selected in the calibration processes. It includes momentum variables of pressure levels of

700 hPa and 850 hPa such as wind-field velocities and changes in vertical pressure and the rainfall prediction on the surface. Additional predictors screening was conducted to finalize a good set of predictors based on "stepwise regression" or known as forward regression.

2.5. Hydrological Model

With respect to hydrological modeling, the tank model is considered a very simple model that has been widely used in rainfall runoff-analysis. This feature is in line with the standpoint of the study that simplified approaches are preferable. However, as a conceptual model, the tank model has many parameters that are required for calibration; it might not be an appropriate selection to assess the rainfall-runoff processes for poor observation basins. The super tank model used in this study aims to overcome this issue. The super tank is also based on the original tank model, being attributed to some physical-base features [16]. Thus, the super tank model is nearly calibration-free requirement, because model parameters are internally calibrated based on catchment geotopographical information. Additionally, the super tank model is semidistributed; therefore, it is assumed to outperform lumped hydrologic models in terms of spatial variation consideration. As a result, the super tank model has demonstrated its robustness and reliability in rainfall-runoff modeling, across a wide range of spatial and temporal scales as described in [2], especially the scarce observation catchments. Evaluation of runoff model performance is based on two criteria, Nash Sutcliffe Index (NSI), or the so-called coefficient of efficiency and the relative error of predicted runoff (η), as expressed in (2.2) and (2.3), respectively

$$\mathrm{NSI} = 1 - \frac{\sum \left(Q_{\mathrm{obs}} - Q_{\mathrm{pred}}\right)^2}{\sum \left(Q_{\mathrm{obs}} - Q_{\mathrm{obs.mean}}\right)^2}, \tag{2.2}$$

$$\eta = \frac{\left|Q_{\mathrm{pred}} - Q_{\mathrm{obs}}\right|}{Q_{\mathrm{obs}}}, \tag{2.3}$$

where, Q_{obs} = observed river flow and $Q_{\mathrm{obs.mean}}$ = mean observed river flow and Q_{pred} = predicted river flow.

3. Results and Discussion

3.1. Downscaling Precipitation

In respect of precipitation prediction, it has been considered as the most difficult variable to be predicted. The precipitation evolution involves a complex process that is not only driven by the dynamic change in atmosphere but also affected by land-surface characteristics. The NWP models, in general, tend to overestimate light rainfall. On the other hand, it seems to severely underestimate intense rainfall, particularly in elevated watersheds. In practice, the longer the forecast lead time is, the most effective flood mitigations can be put in place. Unfortunately, forecast uncertainty is likely to be larger along with the forecast lead time. It is necessary to evaluate to what lead time of the forecast is required for flood mitigation purposes. In the context of this study, forecasting for flood warning was defined. The forecast lead time is supposed to be sufficient, such as for evacuation that is considered within 24 hours, in places like developing countries which have very limited access to proper logistic

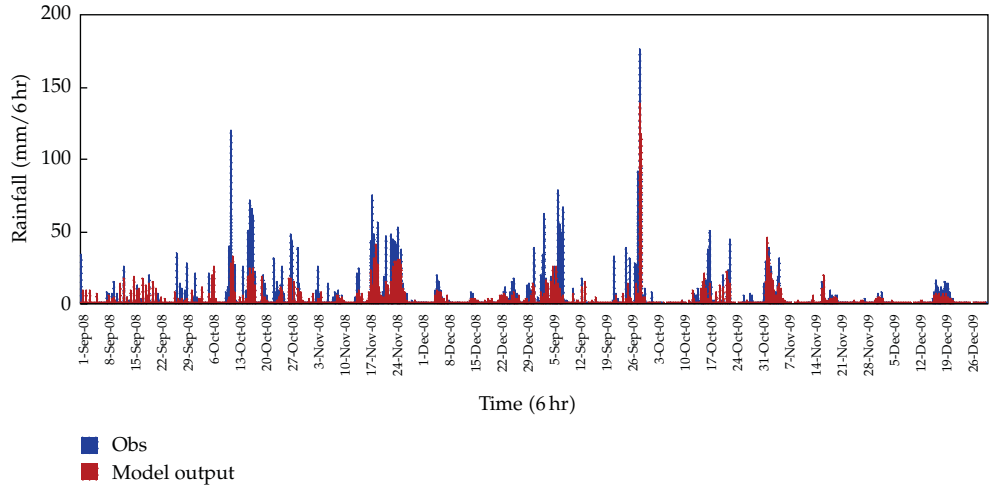


Figure 3: Time series of observed rainfall and those derived from the model outputs for wet seasons, 2008 and 2009.

availability and evacuation routes. As a consequence, analyses are focused on model outputs of 24-hour forecast lead time by updating the exiting forecast lead time (84-hour) on daily basis. In addition to the increased trend of the uncertainty along with the forecast lead time, it has been observed that forecast skill of NWP models is higher as model resolutions are increased.

Preliminary downscaling precipitation was conducted by interpolating the model outputs from the grid point value representation (0.5° or equal to grid cell size of 50 × 50 km) to subgrid scale then was averaged for catchment scale. The comparison of area-average rainfall derived from rain gages and model outputs for periods Sep–Dec, 2008 and 2009, is presented in Figure 3. These periods are classified as wet seasons when most intense rainfall storms were observed. The wet season usually holds up to 70 percent of the total annual precipitation. As seen in Figure 3, model bias was found for most storm events. The model-output-driven precipitation prediction was much lower than the actual observation. A reason for this discrepancy might be explained by the converse behavior of altitudinal dependence of precipitation between actual observation and that obtained from model outputs, especially in elevated watersheds. The comparison of altitudinal dependence of precipitation between observation and model outputs was conducted for various locations (9 rain gages, as seen in Figure 1) with different heights in the Central Vietnam [17]. Results showed that the increase tendency of relationship between the rainfall observed at 9 rain gages and its elevation was found. Rain gages in low location, in other words, close to the shoreline, were observed to show less rainfall than those located in higher elevations. On the contrary, rainfall prediction, which was preliminarily downscaled from model outputs to individual rain-gage using interpolation method, depicted an opposite trend. It was found to be a considerable declining of forecasted rainfall towards the altitudinal increase. The forecasted precipitation was found closer to the observation in low-elevation locations. This model bias might be interpreted as surface elevation is averaged within a coarse spatial resolution; thus, small-scale effects of topography may not be resolved in the global NWP model.

In the following paragraphs, results for downscaling precipitation are presented, using the feed-forward MLP with error back-propagation training algorithm. The input layer of

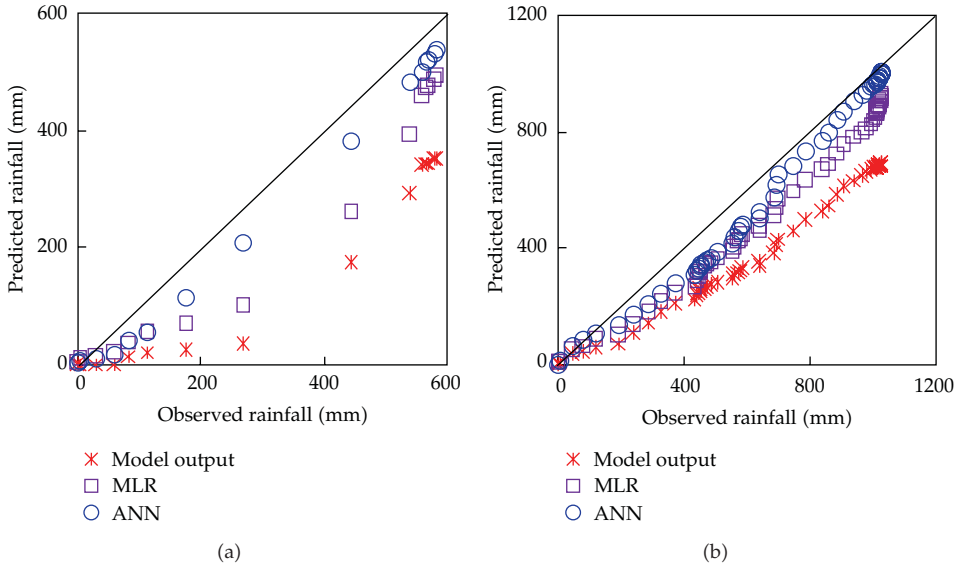


Figure 4: Comparison of accumulative rainfall between observation and prediction obtained from model output and downscale using MLR and ANN for single storm event (a) on Sep. 26–30th, 2009 and continuous storm event (b) on Nov. 17th–27th, 2008.

the feed-forward MLP architecture, as a result of stepwise regression, comprises 3 predictors (3 nodes) that are derived from the model outputs, including vertical changes in atmosphere pressure at (i) the layer 700 hPa, (ii) the layer 850 hPa, and (iii) precipitation prediction on the surface. 6 nodes are selected for the hidden layer. Finally, results from the output layer are the downscaled precipitation. Data of 12 storm events which occurred in wet seasons, 2008 and 2009, was selected for the analysis. The data set was divided into training and validation data, which a majority of studies have used.

The area-average downscaled precipitation and that obtained from model outputs were then compared to the actual area-average observation, known as reference rainfall, on storm event basis. The results showed that there is a low forecast skill of the NWP model at the initial stage of each storm event. The model severely underestimated the actual precipitation. A better forecast skill was found towards the recession limb of the storms. However, the comparison of accumulative precipitation shows that rainfall predicted by the NWP model is much less than the actual observation (Figure 4). These uncertainties can be understood as not only resulting from the averaged surface elevation in a coarse grid cells and the altitudinal bias of the global NWP model but also because of the incompleteness of initial condition assumptions inputted into the NWP models at every initial stage of the storms.

On the other hand, significant improvement of rainfall prediction was observed using the proposed downscaling method. With respect to the comparison of accumulative rainfall between downscale and observation, very good agreements were found. As seen in Figure 4, the downscaled precipitation (circles) shows best fit with the perfect line (solid line). Correlation coefficient for the area-average rainfall increased about 12% and 5% for single storm and continuous storms, respectively. The coefficient of determination (R^2) that

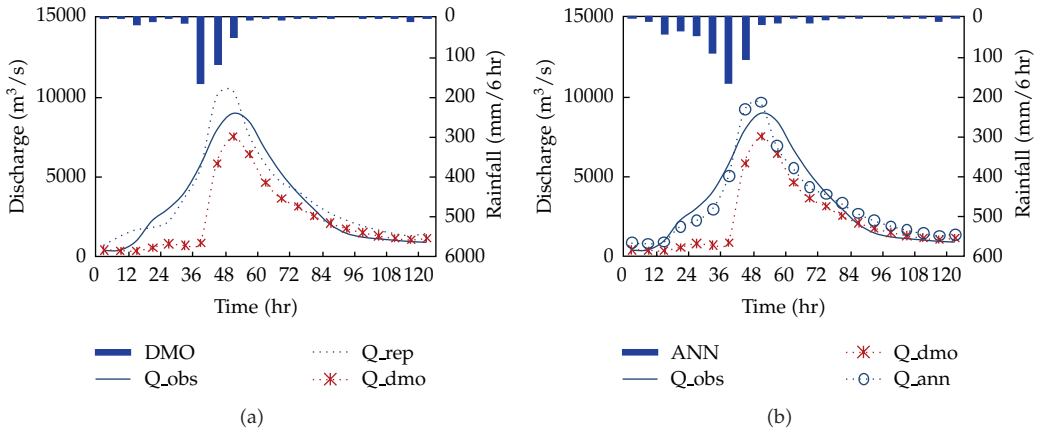


Figure 5: Time series of observed hydrograph (Q_obs) and those based on different precipitation estimation: (a) observed by rain gages (Q_rep) and derived from direct model outputs (Q_dmo), and (b) downscaled using ANN (Q_ann) for the flood event on Sep. 26th–30th, 2009.

measures the fitness of the regression model was found rising from 0.55 for the direct model outputs to 0.96 for the downscaled rainfall using ANN technique for the single storm event. Similarly, values of 0.39 and 0.56 were obtained for the continuous storm event. Meanwhile, skill scores of precipitation forecast based on ANN outperformed those based on DMO, approximately 45% increase. This implies that the present downscaling method attributed with some physical-based features is appropriate, universally applicable across the requirement of a long historical record for the calibration processes. Another regression technique that was presented in [2], the stepwise multiple linear regression (MLR), as expressed in (3.1), is also addressed here for the intercomparison. It was observed that downscaled precipitation using MLR technique showed lower forecast skill score than ANN approach, of about 30%. The R^2 for the downscaled rainfall using MLR technique was found to be 0.68 and 0.46 for the single and continuous events, respectively

$$P_{\text{mlr}} = a_0 + a_1 P_{\text{dmo}} + a_2 X_2 + \cdots + a_n X_n, \tag{3.1}$$

where P_{mlr} is downscaled precipitation using MLR, P_{dmo} is direct model output precipitation forecast, $X_{2,n}$ is independent model output variables, a_o is regression constant, $a_{1,n}$ is regression coefficients, and n is number of independent variables.

3.2. Flood Forecast Based on Downscaled Precipitation

Though the super tank is nearly calibration-free requirement, it is essential to validate parameters of the super tank model through the reproduction of runoff using rainfall information obtained from rain gages. Detailed runoff model setup and validated parameters were presented in [2]. The hydrograph simulated by the super tank model at the outlet of the catchment using input precipitation derived from rain gages, or the so-called reproduction of flood

runoff (Q_rep), for the flood event on September 27–30th, 2009, is plotted in Figure 5(a). It showed a very good agreement with the observed hydrograph (Q_obs). Model performance was evaluated by NSI, showing a very high efficiency, approximately 0.93. Meanwhile, relative error of 16% was observed for the peak discharge.

In next steps, different rainfall estimations from direct model outputs and downscaled results, hereinafter referred to as DMO and ANN, respectively, were then used as inputs to the super tank model to predict flood runoff. Forecasted flood runoff was compared to the observed discharge and also the reproduced flood runoff, as illustrated in Figures 5(a) and 5(b). Considerable uncertainties were observed on the rising limb of the hydrograph that was driven by DMO (Q_dmo). On the other hand, a better forecast towards the recession limb was found. It is evident that the underestimate of precipitation is a main cause for a low hydrograph. It underestimated about 16% lower than the actual peak, but time to peak agreed well with that observed. Overall model performance was indicated by NSI, of about 0.74.

Given the downscaled precipitation was observed for higher forecast skill to the direct model outputs, for that reason, ANN has increased flood forecast skill, as seen in Figure 5(b). The predicted hydrograph (Q_ann) was comparable to that obtained using rain gages (Q_rep), as mentioned in the previous paragraphs. Improved model performance was found, and the model efficiency increased to 0.92; meanwhile, relative error of the peak discharge decreased to 3.8%.

3.3. Model Validation

To evaluate the skill of ANN models, the data set should be ideally divided into three sub-sets, respectively, for learning phase, testing phase, and validating phase [4]. In fact, regarding the limited data availability, only the learning phase and validating phase were conducted. In present study, the storm event on November 1st–7th, 2009 was selected for model validation. Downscaled precipitation was accumulated, and was then compared to that obtained from model outputs as well as actual observation. It was noticed that model-output-driven precipitation remained underestimated, while the downscaled rainfall showed a very good agreement with the actual observation, except a slight overestimate at the end of the storm. These are clearly illustrated in Figure 6. It means that a considerable precipitation prediction skill of ANN model was demonstrated.

The rainfall information was subsequently used to predict the flood runoff. Forecasted hydrographs based on rainfall derived from DMO and downscale using ANN are illustrated in Figure 7. Again, it was found that the DMO-driven hydrograph has severely underestimated the peak discharges. Model efficiency was very low, with NSI of about 0.25 and relative error of the peak of 38%. On the other hand, the downscaled precipitation using ANN-driven flood forecast, in case of this model validation, showed a substantial improvement, approximately 75%, in terms of the model efficiency NSI is up to 0.81. Flood propagation behavior showed very good agreement with the actual observation, especially the time to peak. The model was found to slightly underestimate peak discharges, approximately 25% lower than the actual peaks. However, comparison of total volume showed a very close estimate to the observed volume, just approximately 14% lower than the true volume. In this case, the reliable estimation in the volume of imminent floods will be very useful information that enables the implementation of flood-control measures, for instance, through proper reservoir operation system.

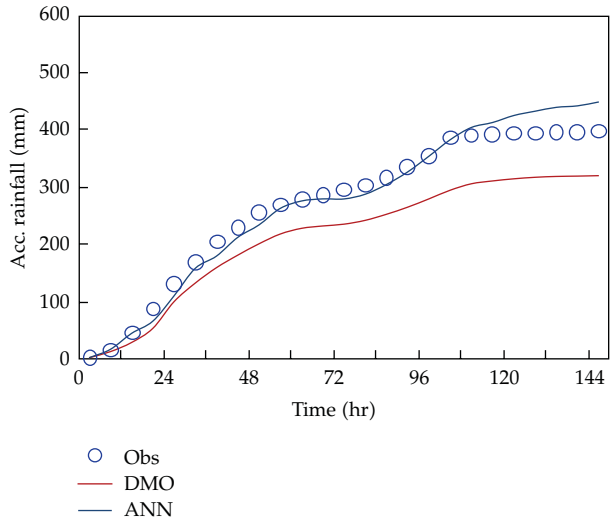


Figure 6: Time series of observed and forecasted hyetographs for the validated storm event on Nov. 1st–7th, 2009.

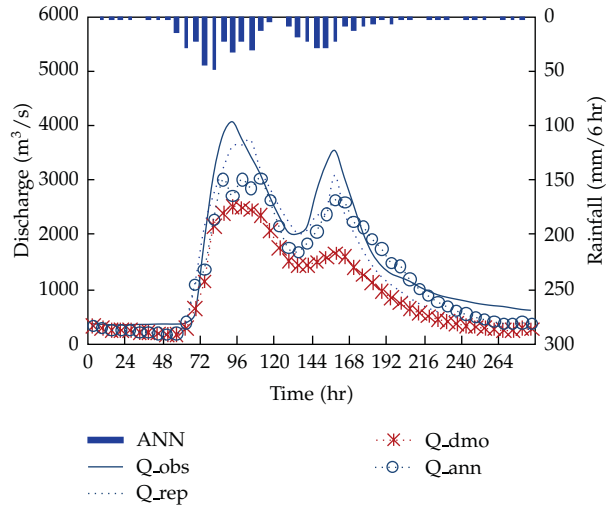


Figure 7: Time series of observed and forecasted hydrographs for the validated flood event on Nov. 1st–7th, 2009.

4. Conclusion

This paper has presented an efficient empirical-statistical approach, using the most favorite ANN architecture, the MLP, with error training back-propagation method, to downscale the precipitation from global NWP outputs to a basin-average scale, subsequently, was used for flood-runoff forecast. The downscaling model has taken into account the physical bases of the precipitation evolution induced by meteorological and land surface characteristics in the study area. As a result, the present model has exhibited cost-effective, simple to implement, and universal application.

The study results indicated considerable uncertainties in precipitation predicted by the global NWP model due to the coarse spatiotemporal resolution and inherent system bias. Accordingly, the flood forecast based on DMO was not what had been expected. It severely underestimated the true hydrograph. By using downscaling approach, however, significant increase of forecast skill was observed for flood prediction based on the downscaled precipitation. The ANN has showed a better learning ability than those using the MLR method.

The presented model has demonstrated the provision of reliable information of the coming flood in a very early stage (24 hr lead time), as considered outperforming other conventional forecasting methods, so that vulnerable communities and flood-control bodies are more active in coping with potential threat and damage in order to insure that flood mitigations are effectively put in place. In addition, it should be stressed that using simple, reliable, and widely applied approaches is the benefit of the study. The prediction model is therefore considered as universally applicable, especially in the developing countries where weather observation is scarce and access to high-resolution weather prediction models is limited.

Acknowledgments

The authors would like to express special thanks to the Institute for International Advanced Research and Education of Tohoku University, Japan, for the financial support of this study. Special thanks are given to Dr. Phil Brierley for provision of MLP neural code and Dr. Jiye Zeng for the grid point value decoding tools.

References

[1] B. Jens and T. Ezio, "Coupling meteorological and hydrological models for flood forecasting," *Hydrology and Earth System Sciences*, vol. 9, no. 4, pp. 333–346, 2005.

[2] D. H. Nam, K. Udo, and A. Mano, "Development of short-term flood forecast model—a case study for central vietnam," *Annual Journal of Hydraulic Engineering, Japan Society of Civil Engineering*, vol. 54, pp. 163–168, 2010.

[3] K. Hsu, H. V. Gupta, and S. Sorooshian, "Artificial neural network modeling of the rainfall-runoff process," *Water Resources Research*, vol. 31, no. 10, pp. 2517–2530, 1995.

[4] C. W. Dawson and R. L. Wilby, "Hydrological modelling using artificial neural networks," *Progress in Physical Geography*, vol. 25, no. 1, pp. 80–108, 2001.

[5] G. Kim and A. P. Barros, "Quantitative flood forecasting using multisensor data and neural networks," *Journal of Hydrology*, vol. 246, no. 1-4, pp. 45–62, 2001.

[6] T. Wardah, S. H. Abu Bakar, A. Bardossy, and M. Maznorizan, "Use of geostationary meteorological satellite images in convective rain estimation for flash-flood forecasting," *Journal of Hydrology*, vol. 356, no. 3-4, pp. 283–298, 2008.

[7] S. Yokoi and J. Matsumoto, "Collaborative effects of cold surge and tropical depression-type disturbance on heavy rainfall in Central Vietnam," *Monthly Weather Review*, vol. 136, no. 9, pp. 3275–3287, 2008.

[8] R. L. Wilby and T. M. L. Wigley, "Downscaling general circulation model output: A review of methods and limitations," *Progress in Physical Geography*, vol. 21, no. 4, pp. 530–548, 1997.

[9] E. B. Rasmus, H. B. Inger, and C. Deliang, *Emprirical-Statistical Downscaling*, World Scientific Publishing, 2008.

[10] E. P. Salathe, "Comparison of various precipitation downscaling methods for the simulation of streamflow in a rainshadow river basin," *International Journal of Climatology*, vol. 23, no. 8, pp. 887–901, 2003.

[11] H. R. Glahn and D. A. Lowry, "The Use of Model Output Statistics (MOS) in objective weather forecasting," *Journal of Applied Meteorology*, vol. 11, pp. 1203–1211, 1972.

[12] J. T. Schoof and S. C. Pryor, "Downscaling temperature and precipitation: a comparison of regression-based methods and artificial neural networks," *International Journal of Climatology*, vol. 21, no. 7, pp. 773–790, 2001.
[13] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, vol. 1, MIT Press, Cambridge, Mass, USA, 1986.
[14] D. Fletcher and E. Goss, "Forecasting with neural networks: an application using bankruptcy data," *Information and Management*, vol. 24, no. 3, pp. 159–167, 1993.
[15] P. Brierley, "Some practical application of neural networks in the electricity industry," Ph.D. thesis, Cranfield University, Cranfield, UK, 1998.
[16] H. Kardhana and A. Mano, "Uncertainty evaluation in a flood forecasting model using JMA-NWP," *Journal of Disaster Research*, vol. 4, no. 4, pp. 272–277, 2009.
[17] D. H. Nam, K. Udo, and A. Mano, "Assessment of altitudinal dependence of rainfall in central vietnam," in *Proceedings of the 12th International Summer Symposium*, Japan Society of Civil Engineering, 2010.