Back-Propagation Learning of Partial Functional Differential Equation with Discrete Time Delay

Tibor Kmet[1] and Maria Kmetova[2]

[1] Constantine the Philosopher University, Department of Informatics,
Tr. A. Hlinku 1, 949 74 Nitra, Slovakia
tkmet@ukf.sk
http://www.ukf.sk
[2] Constantine the Philosopher University, Department of Mathematics,
Tr. A. Hlinku 1, 949 74 Nitra, Slovakia
mkmetova@ukf.sk

Abstract. The present paper describes the back-propagation learning of a partial functional differential equation with reaction-diffusion term. The time-dependent recurrent learning algorithm is developed for a delayed recurrent neural network with the reaction-diffusion term. The proposed simulation methods are illustrated by the back-propagation learning of continuous multilayer Hopfield neural network with a discrete time delay and reaction-diffusion term using the prey-predator system as a teacher signal. The results show that the continuous Hopfield neural networks are able to approximate the signals generated from the predator-prey system with Hopf bifurcation.

Keywords: feed-forward neural network, multilayer Hopfield neural network with a discrete time delay and diffusion-reaction term, time-dependent learning, prey-predator system with a discrete time delay and diffusion-reaction term, numerical solution.

1 Introduction

Partial functional differential equations with a discrete time delay arise in many biological, chemical, and physical systems which are characterized by both spatial and temporal variables and exhibit various spatio-temporal patterns [13]. The recurrent neural network with a time delay and diffusion term are described by a special type of such equations. One of the fundamental ideas of the recurrent neural network is its application to the dynamical memory, a desired attractor embedded into the dynamical system [1],[11]. It is well known that a neural network can be used to approximate the smooth time-invariant functions and the uniformly time-varying function [3], [9]. Besides, it has been used for the universal function approximation to solve optimal control problems with a discrete time delay forward in time to approximate the costate variable [5], [12]. The costate variables play an important role also in the back-propagation learning of infinite-dimensional dynamical systems [10], [11] in the case of time-dependent recurrent learning. Our interest here is in supervised learning of a

G. Agre et al. (Eds.): AIMSA 2014, LNAI 8722, pp. 186–193, 2014.

partial functional Hopfield neural network (PFHNN) with a discrete time delay. The supervised learning is to teach the spatio-temporal dynamics to the PFHNN by applying the back propagation algorithm [8]. In order to learn the dynamic behaviours of the partial functional differential equations based on [10], a time-dependent recurrent learning algorithm has been developed. For the neural network which determines the right hand side of PFHNN, a feed-forward neural network with one hidden layer for the state variable and one hidden layer for the delay state, a steepest descent error backpropagation rule, a hyperbolic tangent sigmoid transfer function and a linear transfer function were used.

This paper is organized as follows. In Section 2 we present a description of the back-propagation learning of infinite-dimensional dynamical systems and propose a new algorithm to calculate the gradient of cost function. Section 3 presents a short description of the prey-predator system and numerical results of the time-dependent recurrent learning using Lagrange multipliers to compute the gradients of the cost function. Conclusions are presented in Section 4.

2 Partial Functional Hopfield Neural Network Learning with a Discrete Time Delay

Let us consider supervised learning to teach the discrete time delay dynamic to the discrete time delay partial functional Hopfield neural network. We utilize the following form of multilayer continuous Hopfield neural network with a discrete time delay in the learning of complex nonlinear dynamics:

$$\begin{aligned}\dot{x}(t) &= D\frac{\partial^2 x(s,t)}{\partial s^2} + F(x(s,t), x(s,t-\tau), W) \\ &= D\frac{\partial^2 x_i(s,t)}{\partial s^2} - Ax(s,t) + W^o f(W^h x(s,t) + W^{hd} x(s,t-\tau)),\end{aligned} \tag{1}$$

with Neumann boundary condition $\frac{\partial x_i}{\partial s}(a,t) = \frac{\partial x_i}{\partial s}(b,t) = 0$ and initial conditions $x_i(s,t) = \phi_i(s,t) \geq 0,\ a \leq s \leq b,\ t \in \langle -\tau, 0\rangle,\ i = 1,\ldots,n$, where $x = (x_1,\ldots,x_n)$, $F = (F_1,\ldots,F_n)$, A_{nxn}, D_{nxn} are diagonal matrices, W^o_{nxn} is a weight matrix between the hidden and output layer, W^h_{nxn}, W^{hd}_{nxn} are the weight matrices between the input and hidden layer and $W = (A, W^h, W^{hd}, W^o)$. Function f is $tanh(.)$ the activation function. t denotes the time, s represents the spatial location and D is a diagonal matrix of the diffusion coefficients. For the given continuous initial condition $\phi(s,t),\ s \in \langle a,b\rangle,\ t \in \langle -\tau, 0\rangle$ there exists a unique solution $x(s,t)$ satisfying Eq. (1) for $s \in \langle a,b\rangle,\ t \in \langle 0,T\rangle$. The aim is to find the weight parameters W that give rise to a solution $x(s,t)$ approximately following the teacher signal $p(s,t) = (p_1(s,t),\ldots,p_n(s,t))$, where $p(s,t)$ is a solution of the following delay partial functional differential equation:

$$\dot{p}(s,t) = D\frac{\partial^2 p(s,t)}{\partial s^2} + G(p(s,t), p(s,t-\tau)), \tag{2}$$

with Neumann boundary condition $\frac{\partial p_i}{\partial s}(a,t) = \frac{\partial p_i}{\partial s}(b,t) = 0$ and initial conditions $p_i(s,t) = \phi_i(s,t) \geq 0,\ a \leq s \leq b,\ t \in \langle -\tau, 0\rangle,\ i = 1,\ldots,n$. First, the cost function is defined for the weight parameters W as

$$E(W) = \int_a^b \int_0^T \frac{1}{2} \sum_{i=1}^{n} (x_i(s,t) - p_i(s,t))^2 dtds. \tag{3}$$

Then the cost function (3) is minimized by the steepest descent method

$$w^{j+1} = w^j - \alpha \frac{\partial E}{\partial w}(W^j), \tag{4}$$

where $w \in W$. To compute the gradient of function (3), we use time-dependent recurrent learning (TDRL) [11]. In the TDRL algorithm, the gradients are computed by using the Lagrange multipliers $\lambda(t) = (\lambda_1(s,t),\ldots,\lambda_n(s,t))$. For a detailed explanation, see [11]. We can rewrite the cost function $E(W)$ as

$$\begin{aligned} L(W) = \int_a^b \int_0^T \sum_{i=1}^{n} [\frac{1}{2}((x_i(s,t) - p_i(s,t))^2 - \\ \lambda_i(s,t)(\dot{x}_i(s,t) - d_i \frac{\partial^2 x_i(s,t)}{\partial s^2} - F_i(x(s,t), x(s,t-\tau), W))] dtds. \end{aligned}$$

The partial derivatives with respect to the weight coefficients $w \in W$ are calculated as

$$\begin{aligned} \frac{\partial L}{\partial w} = \int_a^b \int_0^T \sum_{i=1}^{n} [(x_i(s,t) - p_i(s,t)) \frac{\partial x_i(s,t)}{\partial w} - \lambda_i(s,t) \frac{\partial \dot{x}_i(s,t)}{\partial w} + \\ \lambda_i(s,t) \frac{F_i(x(s,t), x(s,t-\tau), W)}{\partial w} + \\ \lambda_i(s,t) \sum_{j=1}^{n} \frac{F_i(x(s,t), x(s,t-\tau), W)}{\partial x_j(s,t)} \frac{\partial x_j(s,t)}{\partial w} + \\ \lambda_i(s,t) \sum_{j=1}^{n} \frac{F_i(x(s,t), x(s,t-\tau), W)}{\partial x_j(s,t-\tau)} \frac{\partial x_j(s,t-\tau)}{\partial w} + \\ \lambda_i(s,t) \frac{\partial}{\partial w} \left(d_i \frac{\partial^2 x_i(s,t)}{\partial s^2} \right) - \\ \frac{\lambda_i(s,t)}{\partial w} (\dot{x}_i(s,t) - d_i \frac{\partial^2 x_i(s,t)}{\partial s^2} F_i(x(s,t), x(s,t-\tau), W))] dtds. \end{aligned} \tag{5}$$

If $x(s,t)$ is a solution of Eq. (1) then the final term of Eq. (5) vanishes. Since $\frac{\partial x(s,t)}{\partial w} = 0$ for $t \in \langle -\tau, 0\rangle$ the fourth term of (5) can be written by the transformation $t' = t - \tau$ as

$$\int_a^b \int_0^T \sum_{i=1}^n \lambda_i(s,t) \sum_{j=1}^n \frac{F_i(x(s,t),x(s,t-\tau),W)}{\partial x_j(s,t-\tau)} \frac{\partial x_j(s,t-\tau)}{\partial w} dtds =$$
$$\int_0^T \int_{-\tau}^{T-\tau} \sum_{i=1}^n \lambda_i(s,t+\tau) \sum_{j=1}^n \frac{F_i(x(s,t+\tau),x(s,t),W)}{\partial x_j(s,t)} \frac{\partial x_j(s,t)}{\partial w} dtds =$$
$$\int_0^T \int_0^T \sum_{i=1}^n \lambda_i(s,t+\tau) \sum_{j=1}^n \frac{F_i(x(s,t+\tau),x(s,t),W)}{\partial x_j(s,t)} \frac{\partial x_j(s,t)}{\partial w} \chi_{\langle 0,T-\tau\rangle} dtds.$$

Using intergation by parts with the Neumann boundary condition $\partial\lambda_i(a,t)/\partial s = \partial\lambda_i(b,t)/\partial s = 0$ we get

$$\int_a^b \lambda_i(s,t) \frac{\partial}{\partial w} \left(d_i \frac{\partial^2 x_i(s,t)}{\partial s^2} \right) ds = \int_a^b d_i \frac{\partial^2 \lambda_i(s,t)}{\partial s^2} \frac{\partial x_i(s,t)}{\partial w} ds.$$

The derivatives $\frac{\partial L}{\partial w}$ become

$$\begin{aligned} \frac{\partial L}{\partial w} = \int_a^b \int_0^T \sum_{i=1}^n [(x_i(s,t) - p_i(s,t)) \frac{\partial x_i(s,t)}{\partial w} - \lambda_i(s,t) \frac{\partial \dot{x}_i(s,t)}{\partial w} + \\ \lambda_i(s,t) \frac{F_i(x(s,t),x(s,t-\tau),W)}{\partial w} + d_i \frac{\partial^2 \lambda_i(s,t)}{\partial s^2} \frac{\partial x_i(s,t)}{\partial w} + \\ \lambda_i(s,t) \sum_{j=1}^n \frac{F_i(x(s,t),x(s,t-\tau),W)}{\partial x_j(s,t)} \frac{\partial x_j(s,t)}{\partial w} + \\ \lambda_i(s,t+\tau) \sum_{j=1}^n \frac{F_i(x(s,t+\tau),x(s,t),W)}{\partial x_j(s,t)} \frac{\partial x_j(s,t)}{\partial w} \chi_{\langle 0,T-\tau\rangle}] dtds. \end{aligned} \tag{6}$$

The Lagrange multipliers are solutions of the following partial functional differential equations with the Neumann boundary condition $\partial\lambda_i(a,t)/\partial s = \partial\lambda_i(b,t)/\partial s = 0$ and terminal condition $\lambda(s,T) = 0$.

$$\begin{aligned} -\dot{\lambda}_i(s,t) = d_i \frac{\partial^2 \lambda_i(s,t)}{\partial s^2} + \sum_{j=1}^n \lambda_j(t) \frac{F_j(x(s,t),x(s,t-\tau),W)}{\partial x_i(s,t)} + \\ \sum_{j=1}^n \lambda_j(s,t+\tau) \frac{F_j(x(s,t+\tau),x(s,t),W)}{\partial x_i(s,t)} \chi_{\langle 0,T-\tau\rangle} + (x_i(s,t) - p_i(s,t)). \end{aligned} \tag{7}$$

Since the Lagrange multipliers $\lambda(s,t)$ satisfy Eq. (7) with Neumann boundary condition $\partial\lambda_i(a,t)/\partial s = \partial\lambda_i(b,t)/\partial s = 0$ and terminal condition $\lambda(s,T) = 0$, and $\frac{\partial x(s,t)}{\partial w} = 0$ for $t \in \langle -\tau, 0\rangle$ all the terms of Eq. (6) but the third vanish. The partial derivatives $\frac{\partial J}{\partial w}$ can be calculated by the following form:

$$\frac{\partial L}{\partial w} = \int_a^b \int_0^T \sum_{i=1}^n \lambda_i(s,t) \frac{F_i(x(s,t),x(s,t-\tau),W)}{\partial w} dtds. \tag{8}$$

Algorithm 1. Time dependent recurrent learning algorithm to determine the weight matrix of a time delay continuous Hopfield neural network

Input: Choose T, τ, $p(s,t)$ - teacher signal, $maxit$, ε_E, - stopping tolerance, $\phi(s,t)$, $t \in \langle -\tau, 0\rangle$, $s \in \langle a,b\rangle$, - initial condition.

Output: Weight matrix $\mathbb{W} = (A, W^o, W^h, W^{dh})$;

1 Set the initial weight $\mathbb{W} = (A, W^o, W^h, W^{dh})$, $i = 0$

while $err_E \geq \epsilon_E$ ***and*** $i \leq$ $maxit$ **do**

2 Compute solution $x(s,t)$ of Eq. (1) on the interval $\langle 0, T\rangle$ with initial condition $\phi(s,t), t \in \langle -\tau, 0\rangle$, $s \in \langle a,b\rangle$

3 Compute solution $\lambda(s,t)$ of Eq. (7) on the interval $\langle T, 0\rangle$ with terminal condition $\lambda(s,T) = 0$

4 Compute $E(W)$ by Eq. (2)

5 Compute $\frac{\partial L}{\partial W} = \int_a^b \int_0^T \sum_{i=1}^n \lambda_i(s,t) \frac{F_i(x(s,t),x(s,t-\tau),W)}{\partial W} dtds$ by Eq. (8)

6 Compute $\alpha^* = min\ g(\alpha) = E\left(W^i - \alpha \frac{\partial J(W^i)}{\partial W}\right)$

7 Set $W^{i+1} = W^i - \alpha^* \frac{\partial L(W^i)}{\partial W}$

8 Compute $E(W^{i+1})$ by Eq. (2)

9 Set $err_E = abs(E(W^{i+1} - E(W^i))$

10 **return** $\mathbb{W}^{i+1} = (A^{i+1}, W^{o,i+1}, W^{h,i+1}, W^{dh,i+1})$

We can state the following algorithm for time dependent recurrent learning. To find the minimizer weight matrix W using the Algorithm 1 we can also use the Fletcher-Reeves, DFP and BFGS methods [7]. For the PFHNN Eq. (1) the gradients are calculated as:

$$\frac{\partial L}{\partial a_{ii}} = \int_a^b \int_0^T x_i(s,t)\lambda_i(s,t)dtds, \quad \frac{\partial L}{\partial w^o_{ij}} = \int_a^b \int_0^T \lambda_i(s,t) f_j(s,t) dtds$$

$$\frac{\partial L}{\partial w^h_{ij}} = \int_a^b \int_0^T \sum_{k=1}^n \lambda_k(s,t) w^o_{ki} f_i'(s,t) x_j(s,t) dtds,$$

$$\frac{\partial L}{\partial w^{hd}_{ij}} = \int_a^b \int_0^T \sum_{k=1}^n \lambda_k(s,t) w^o_{ki} f_i'(s,t) x_j(s,t-\tau) dtds,$$

where $f_j(s,t) = tanh\left(\sum_{k=1}^n (w^h_{jk} x_k(s,t) + w^{hd}_{jk} x_k(s,t-\tau))\right)$.

The derivatives $\partial L/\partial \omega$ are computed by the discretization of the diffusion term by finite dimensional approximation [6]. The resulting semidiscrete approximation of (1) amounts to an $N \times N$ system of delay differential equations. The delay differential equations are integrated by the Euler methods using linear spline approximation described in [2].

3 Discrete Time Delay Prey-Predator Ecological Model

This section presents experimental studies of applying the time-dependent learning algorithm developed in Section 2. Let us consider the following prey-predator model [4] with the discrete time delay τ and diffusion term.

$$\begin{aligned}
\dot{p}_1(s,t) &= d_1 \frac{\partial^2 p_1(s,t)}{\partial s^2} + r p_1(s,t)(1 - p_1(s,t)/K) - \sigma_1 p_1(s,t) + \sigma_2 p_2(s,t), \\
\dot{p}_2(s,t) &= d_2 \frac{\partial^2 p_2(s,t)}{\partial s^2} + z p_2(s,t)(1 - p_2(s,t)/L) + \sigma_1 p_1(s,t) - \sigma_2 p_2(s,t) - \\
&\quad \frac{\alpha p_2(s,t) p_3(s,t)}{a + p_2(s,t)}, \\
\dot{p}_3(s,t) &= d_3 \frac{\partial^2 p_3(s,t)}{\partial s^2} + \frac{\beta \alpha p_2(s,t-\tau) p_3(s,t-\tau)}{a + p_2(s,t-\tau)} - d p_3(s,t) - \gamma p_3(s,t)^2,
\end{aligned} \tag{9}$$

with Neumann boundary condition

$$\frac{\partial p_i}{\partial s}(a,t) = \frac{\partial p_i}{\partial s}(b,t) = 0 \tag{10}$$

and initial conditions

$$p_i(s,t) = \phi_i(s,t) \geq 0,\ a \leq s \leq b,\ t \in \langle -\tau, 0 \rangle,\ i = 1, \ldots, 3, \tag{11}$$

where $p_1,\ p_2$ denote the prey populations and p_3 is a predator population. The

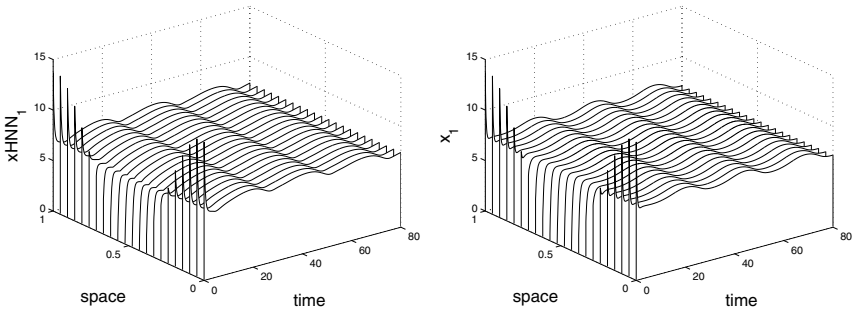

Fig. 1. Numerical solution of the prey-predator model described by Eq. (9) (right part) and numerical solution of the PFHNN Eq. (1) (left part) for $\tau = 3.7$, with teacher initial condition $\phi_1(s,t) = 7(1 + cos(2\pi s))$, $\phi_2(s,t) = 3(1 + cos(2\pi s))$, $\phi_3(s,t) = 2(1 + cos(2\pi s))$, for $t \in \langle -\tau, 0 \rangle$.

equilibria of the model were determined and the behavior of the system was investigated around the equilibria in [4]. Jana et al. [4] obtain that the time delay can cause a stable equilibrium to become unstable and even a simple Hopf bifurcation occurs when the time delay passes through its critical value. The prey-predator model (9) was numerically analyzed in [4] for the given set of parameters ($r = 0.9,\ K = 9,\ \sigma_1 = 0.2,\ \sigma_2 = 0.15,\ z = 0.8,\ L = 14,\ \alpha = 2.5,\ a = 1.2,\ \beta = 0.32,\ d = 0.3,\ \gamma = 0.1$ It was obtained that the solution of Eq. (9) for $\tau = 0.5$ converges to an equilibrium point for $\tau = 3.7$ Eq. (9) has periodic solution [4]. We can use the periodic solution $p(s,t)$ of Eq. (9) for

$\tau = 3.7$ as teacher signals to verify Alg. 1. After 1896-iterative learning the following network weight matrix was obtained for $\tau = 3.7$:

$$W^{(\tau=3.7)} = \left(A, W^o, W^h, W^{hd}\right) =$$

$$\left(\begin{array}{ccc|ccc|ccc|ccc} 0.124 & 0 & 0 & 0.6976 & 0.3013 & 0.1362 & 0.0587 & 0.0203 & 0.0128 & 0.4667 & -0.0324 & 0.1636 \\ 0 & 0.0728 & 0 & -1.2416 & 1.7255 & 1.4959 & -0.1268 & 0.2905 & 0.0064 & 0.5857 & -0.8518 & -0.2421 \\ 0 & 0 & 0.1591 & 0.3649 & -0.1866 & 1.0633 & 0.0386 & -0.0933 & -0.1187 & 0.0794 & 0.0771 & -0.3198 \end{array}\right).$$

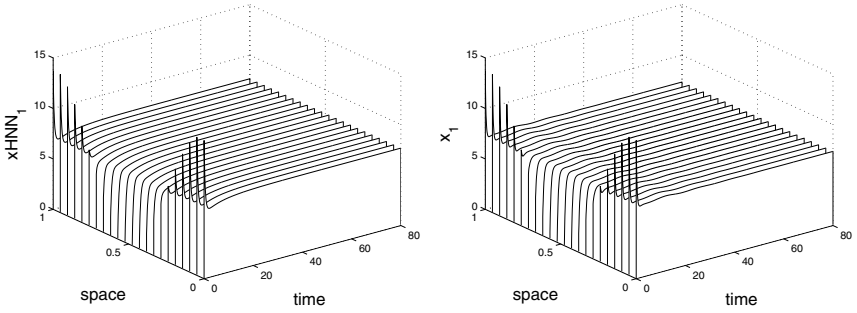

Fig. 2. Numerical solution of the prey-predator model described by Eq. (9) (right part) and numerical solution of the PFHNN Eq. (1) (left part) for $\tau = 0.5$, with the initial condition $\phi_1(s,t) = 7(1 + cos(2\pi s))$, $\phi_2(s,t) = 3(1 + cos(2\pi s))$, $\phi_3(s,t) = 2(1 + cos(2\pi s))$, for $t \in \langle -\tau, 0\rangle$ and weight matrix $W^{(\tau=3.7)}$.

The numerical solutions are shown in Figs. 1, 2. The PFRNN achieved qualitatively similar dynamics in the original predator-prey model. For the $\tau = 0.5$ solutions of Eq. (1) and Eq. (9) converge to the equilibrium point. If $\tau = 3.7$ then we obtain periodic solutions for both systems. It follows from Figs. 1, 2 that the proposed discrete time delay continuous Hopfield neural network is able to approximate the discrete time delay partial functional differential equations.

4 Conclusion

The purpose of the paper is to develop an efficient time-dependent recurrent learning algorithm to determine the weight matrix of the discrete time delay partial functional Hopfield neural network. A signal generated from the simple predator-prey model is used as a learning example. Depending on the discrete time delay τ, the Hopf bifurcation incurred into the system for a given set of parameters of the model. The MATLAB simulations show that the proposed time-dependent learning algorithm is able to determine the weight matrix W and the partial functional Hopfiel neural network gives a good approximation of the predator-prey model.

Acknowledgment. The present paper was made possible thanks to the scientific project VEGA 1/0699/12.

References

1. Becerikli, Y., Konar, A.F., Samad, T.: Inteligent optimal control with dynamic neural networks. Neural Network 16, 251–259 (2003)
2. Farmer, J.D.: Chaotic Attractors of an infinite-dimensional dynamic system. Physica 4D, 366–393 (1982)
3. Hornik, M., Stichcombe, M., White, H.: Multilayer feed forward networks are universal approximators. Neural Networks 3, 256–366 (1989)
4. Jana, S., Chakraborty, M., Chakraborty, K., Kar, T.K.: Global stability and bifurcation of time delayed prey-predator system incorporating prey refuge. Mathematics and Computers in Simulation 85, 57–77 (2012)
5. Kmet, T., Kmetova, M.: Adaptive Critic Neural Network Solution of Optimal Control Problems with Discrete Time Delays. In: Mladenov, V., Koprinkova-Hristova, P., Palm, G., Villa, A.E.P., Appollini, B., Kasabov, N. (eds.) ICANN 2013. LNCS, vol. 8131, pp. 483–494. Springer, Heidelberg (2013)
6. Morton, K.W.: Numerical solution of Convection-Diffusion Problems. Chapman - Hall, London (1996)
7. Polak, E.: Optimization Algorithms and Consistent Approximation. Springer, New York (1997)
8. Rumelhart, D.F., Hinton, G.E., Wiliams, R.J.: Learning internal representation by error propagation. In: Rumelhart, D.E., McClelland, D.E., PDP Research Group (eds.) Parallel Distributed Processing: Foundation, vol. 1, pp. 318–362. The MIT Press, Cambridge (1987)
9. Sandberg, E.W.: Notes on uniform approximation of time-varying systems on finite time intervals. IEEE Transactions on Circuits and Systems-1: Fundamental Theory and Applications 45(8), 305–325 (1998)
10. Tokuda, I., Hirai, Y., Tokunaga, R.: Back-propagation learning of infinite-dimensional dynamical system. In: Proceedings of 1993 International Conference on Neural Network, pp. 2271–2275 (1993)
11. Tokuda, I., Tokunaga, R., Aihara, K.: Back-propagation learning of infinite-dimensional dynamical systems. Neural Networks 16, 1179–1193 (2003)
12. Werbos, P.J.: Approximate dynamic programming for real-time control and neural modelling. In: White, D.A., Sofge, D.A. (eds.) Handbook of Intelligent Control: Neural Fuzzy, and Adaptive Approaches, pp. 493–525. Van Nostrand Reinhold, New York (1992)
13. Wu, J.: Theory and Applications of Partial Functional Differential Equations. Applied Math. Sciences, vol. 119. Springer, New York (1996)