The Problem of Stability in the Application of Neural Network to Continuous-Time Dynamic Systems

Tae-dok Eom, Sung-Woo Kim, Kang-Bark Park and Ju-Jang Lee
Department of Electrical Engineering
Korea Advanced Institute of Science and Technology
373-1 Kusong-dong Yusong-gu
Taejon 305-701 Korea
email: jjlee@ee.kaist.ac.kr

Abstract

Using neural network to identify a function in the dynamic equation brings about additional difficulties which are not generic in the other function approximation problems. First, training samples can not be arbitrarily chosen due to hard nonlinearity, so are apt to be nonuniform over the region of interest. Second, the system may become unstable while attempting to obtain the samples.

This paper deals with these problems in continuous-time systems and suggest an effective solution, which provides stability and uniform sampling by the virtue of supervisory controller. The supervisory control algorithm can be applied to the robot system dynamics. Of course the algorithm can be applied to the n-th order robot system, the simulation result will be given for the simple two link robot.

1 Introduction

In recent years, neural network has found wide applications in areas such as system identification, modeling and realization, signal decomposition and generation, pattern classification, adaptive filtering, *etc.* One of the most significant backgrounds of these trends is the function approximation capability of neural network. The problem of approximating a function of several variables by neural network has been studied by many authors. They proved various kind of neural network structure can be an universal approximator of the functions in $C[D]$, the set of continuous real-valued functions on a compact space D [1], and in $L^p[D]$, consisting of all real measurable Lebesque-integrable functions[3].

Besides the classical problems of neural network mentioned above, additional problems arise in the applications for dynamic systems. In the early days of neural network, the applications were primarily in the area of pattern recognition and hence pertains to static systems. Since dynamics constitute an essential part of all practical systems, it was tried by many authors [4] to use neural networks as components in dynamical systems. The various designs of control architectures are studied and extensive simulations are carried out to show the models proposed are particularly effective for the identification and control of nonlinear systems. However, much of the work is of a heuristic nature. Narendra *et al.* [5] presented a first attempt to relate the experimental studies to theoretical developments and tried to propose a general methodology by which control methods based on neural networks can be made more rigorous.

In most of the papers which seek for the general methodology in using neural networks for dynamic systems, the efforts are mainly contributed to the stability analysis. These analyses are based on the assumptions endowed to systems, in the other words, if neural network is applied to a design methodology, the function in the system should be examined whether it satisfies the prescribed assumptions or not. But, it does not seem always possible to judge the satisfaction of such assumptions proposed by Narendra [5]. Hence, it is needed to relax the stringent assumptions to the simplified ones such as the boundedness of the function, *etc.*

This paper deals with the problem of uniform sampling as well as the stability problem in certain class of continuous time systems. The problem of the uniform sampling is issued by Narendra [6]. They proposed the successive identification and control strategy to solve the problem for certain class of discrete-time system. We adopted the supervisory controller which Wang [7] developed for his fuzzy adaptive controller. And we modified it to apply to the robot dynamics together with neural network identifier. If the function in the dynamic equation is bounded, the supervisory controller guarantees the bounded tracking error. Therefore, once the function in the dynamic equation is identified along a uniform state trajectory preserving the stability with the supervisory controller, any tracking control along the same path can be successfully performed.

2 Supervisory Control Algorithm

In this section, supervisory control algorithm is introduced to provide the bounded tracking error while exploring the region of interest along the path designated by the desired trajectory, $x_d(t)$. Cause a term

0-8186-7108-4/95 $4.00 © 1995 IEEE

$u_c(t)$ is remained undesigned in supervisory control algorithm, any kind of control scheme can be adopted to generate $u_c(t)$. We use feedforward neural network to identify a function which handles the system dynamics. With this identifier, classic PD-type controller is also working in $u_c(t)$ to help the supervisory controller.

The target system equation for which supervisory control algorithm will be designed is given by the equation of motion

$$x^{(n)} = f(\mathbf{x}) + bu \tag{1}$$

where $x^{(n)}$ is the n-th derivative of x, $\mathbf{x} = [x, \dot{x}, \cdots, x^{(n-1)}]^T$, $x, b, u \in \Re$, and f is an unknown mapping.

And the followings are assumed to be valid in our analysis.

Assumption 1 *The function f in* (1) *should be bounded,*

$$|f(\mathbf{x})| \leq f^U(\mathbf{x}) \; for \, \forall \mathbf{x} \in \Re^n \tag{2}$$

Assumption 2 *The state vector $\mathbf{x} = [x, \dot{x}, \cdots, x^{(n-1)}]^T$ in* (1) *is measurable.*

Because supervisory control algorithm is needed to assure the boundness of tracking error, (1) should be changed into the dynamics of tracking error. $\mathbf{x}_d$ is the desired trajectory and let the error vector $\mathbf{x}_e = \mathbf{x} - \mathbf{x}_d$, then the system can be controlled with the following.

$$u^* = \frac{1}{b}[-f(\mathbf{x}) + x_d^{(n)}(t) - \mathbf{k}^T\mathbf{x}_e] \tag{3}$$

where the gain $\mathbf{k} = [k_1, k_2, \cdots, k_n]^T$ is chosen to make the polynomial $h(s) = s^{(n)} + k_n s^{(n-1)} + \cdots + k_1$ be a Hurwitz. Applying (3) to (1) renders the system asymptotically stable, but this control input can not be implemented since $f(\mathbf{x})$ in u^* is unknown. This impractical input, u^*, just helps the conversion of (1) to tracking error dynamics.

The design of the supervisory controller is based on Lyapunov's direct method. Suppose that the input, u, is the addition of the neural network controller, u_c, which will be designed later, and the supervisory controller, u_s, that is

$$u = u_c + u_s \tag{4}$$

Substituting (4) to (1), (1) becomes

$$x^{(n)} = f(\mathbf{x}) + b(u_c + u_s) \tag{5}$$

By subtracting bu^* from both sides of (5), the equation of error is obtained:

$$x_e^{(n)} = -\mathbf{k}^T\mathbf{x}_e + b(u_c + u_s - u^*) \tag{6}$$

or equivalently in vector form,

$$\dot{\mathbf{x}}_e = \Lambda\mathbf{x}_e + \mathbf{b}(u_c + u_s - u^*) \tag{7}$$

where Λ and $\mathbf{b}$ are the controllable canonical form matrices. Define $V = \frac{1}{2}\mathbf{x}_e^T P \mathbf{x}_e$ where P is a symmetric positive definite matrix satisfying the following Lyapunov equation.

$$\Lambda^T P + P\Lambda = -Q \tag{8}$$

where Q is a positive definite matrix.

Now, the following theorem is derived based on the Lyapunov's direct method.

Theorem 1 *Consider the system described by* (1), *satisfying* Assumption 1 − 2, *and subject to control given by*

$$u = u_c + u_s \tag{9}$$

and

$$\begin{aligned} u_s \;&=\; -Isgn(\mathbf{x}_e^T P\mathbf{b})[|u_c| \\ &\quad + \frac{1}{b}(f^U + |x_d^{(n)}| + |\mathbf{k}^T\mathbf{x}_e|)] \end{aligned} \tag{10}$$

where $I = 1$ if $V > V_M$ and $I = 0$ otherwise, and $V_M > 0$ is a constant specified by the designer.
Then, $V < V_M$ as $t \to \infty$.

Proof The derivative of V along the system trajectory becomes

$$\begin{aligned} \dot{V} \;&=\; -\frac{1}{2}\mathbf{x}_e^T Q\mathbf{x}_e + \mathbf{x}_e^T P\mathbf{b}(u_c + u_s - u^*) \\ &\leq\; -\frac{1}{2}\mathbf{x}_e^T Q\mathbf{x}_e + |\mathbf{x}_e^T P\mathbf{b}|(|u^*| + |u_c|) \\ &\quad + \mathbf{x}_e^T P\mathbf{b}u_s \end{aligned} \tag{11}$$

By the supervisory control, u_s, when $V > V_M$, $\dot{V}$ becomes

$$\begin{aligned} \dot{V} \;&\leq\; -\frac{1}{2}\mathbf{x}_e^T Q\mathbf{x}_e + |\mathbf{x}_e^T P\mathbf{b}|[\frac{1}{b}(|f| - f^u)] \\ &\leq\; -\frac{1}{2}\mathbf{x}_e^T Q\mathbf{x}_e < 0 \end{aligned} \tag{12}$$

So, $V < V_M$ as $t \to \infty$.

□

u_s can be implemented with the knowledge of $|u_c|$ which are not specified in the context of the proof. Therefore any kind of controller can be a candidate for u_c. And, the supervisory control u_s in Theorem 1 is zero, *i.e.*, $u = u_c$, if V is smaller than V_M. So to speak, u_s takes a rest as long as u_c does his job well, but becomes activated to push the error vector inside the bound, if u_c fails to make $V < V_M$.

3 Identification and Control Using SCA

Instead of the infeasible control input u^* of (3), the controller which consists of neural network identifier and PD-type controller is used in our identification procedure:

$$u_c = \frac{1}{b}(-N_f + x_d^{(n)} - \mathbf{k}^T \mathbf{x}_e) \qquad (13)$$

where N_f takes the place of f in (3). Of course any kind of control scheme is available for the candidate of u_c, controller of (13) with simple FNN in it is chosen in this paper. It is also obvious that with the help of supervisory controller, u_s, the system controlled by u_c does not lose stability even in the case the shape of N_f is far from that of f.

Another merit of using supervisory controller is the fact that neural network is trained within the specified region about the desired trajectory. The identification along the desired trajectory is very efficient for the task of following the desired trajectory. The uniform sampling over the range of neural network input, can not be achieved if we arbitrarily choose the desired trajectory. Generally in tracking control of high order systems, the uniform sampling is not available. But the modified supervisory control algorithm which will be suggested in the next section, can guarantee the uniform sampling for the regulation control of robot dynamics.

As iteration goes over and over, N_f is trained by the samples from f with back propagation algorithm, and finally cancels out f. If the networks becomes sufficiently accurate to verify the cancellation, then the error is described by

$$x_e^{(n)} + k_n x_e^{(n-1)} + \cdots + k_1 x_e \cong 0 \qquad (14)$$

, so that the system is controlled to be asymptotically stable.

Of course if we set V_M to zero, the error is not only bounded but also asymptotically stable. However, this control input resembles the relay control of VSC, so control effort is excessive and the chattering occurs. Combining the supervisory control and PD-type control with neural network identifier, such problems can be eliminated.

Simulation : In this example, we consider the Duffing forced oscillation system:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -0.1x_1 - x_1^3 + 12\cos(t) + u(t) \end{aligned} \qquad (15)$$

Without control, *i.e.*, $u(t) = 0$, the system is chaotic. The trajectory of the system with $u(t) = 0$ is shown in the (x_1, x_2) phase plane in Fig.1 for initial condition $x_1(0) = x_2(0) = 2$. We now use our supervisory control algorithm to track the reference trajectory $x_d(t) = \sin(t)$. We choose $k_1 = 1$, $k_2 = 2$, $Q = diag[10, 10]$, $V_M = 1$, and $f^U = 12 + |x_1|^3$.

Of course the system described by equation (15) is not autonomous. f is a function of $\mathbf{x}$ and t. Therefore we choose the remainder when t is divided by the revolution period as one of the input variables to N_f. Fig.2 shows the tracking control along the unit circle at the revolution of 1, 200, 1000 times. Fig.3 depicts the approximation of f by N_f at the first revolution. In Fig.4, the approximation is close enough to cancel the nonlinearty, so the system is no longer nonlinear system.

In our supervisory control algorithm, it is necessary to select a sufficiently small V_M. If V_M is too big, the path which can be taken maintaining $V < V_M$, is diverse that the identification is not efficient.

4 Application of the SCA to Robot Dynamics

The robot dynamics can be described by the following equation.

$$M(q)\ddot{q} + N(q, \dot{q}) = \tau \qquad (16)$$

where $M(q)$ is an inertia matrix and $N(q, \dot{q})$ includes centrifugal, Coriolis, and gravitational force. To apply the supervisory control algorithm, the following assumptions should be valid for the dynamics described by equation (16).

Assumption 3 *Inertia matrix $M(q)$ is known, i.e., all uncertainties belong to $N(q, \dot{q})$.*

Assumption 4 *The state vector $q, \dot{q}$ in (16) is measurable.*

Besides the assumptions above, robot dynamics has following property.

Property 1 In robot dynamics described by (16), there exist $c_0, c_1 > 0$ such that

$$N(q, \dot{q}) \leq c_0 + c_1 \|\dot{q}\|^2. \qquad (17)$$

Define $F(q, \dot{q}) = -M^{-1}(q)N(q, \dot{q})$ and $u = M^{-1}\tau$, then the equation (16) is changed to,

$$\ddot{q} = F(q, \dot{q}) + u. \qquad (18)$$

To induce the tracking error dynamics, define q_d is the desired trajectory, and let the error vector $q_e = q - q_d$, then the robot system can be controlled with the following control input.

$$u^* = -F(q, \dot{q}) + \ddot{q}_d - Ke \qquad (19)$$

where $e^T = [q_e^T \ \dot{q}_e^T]$ is the new error vector and $K = [K_1 \ K_2]$ is chosen to make the matrix

$$\begin{bmatrix} 0 & I \\ -K_1 & -K_2 \end{bmatrix} \qquad (20)$$

have the eigenvalues in the left half plane. Applying (19) to (18) renders the system asymptotically stable, but this control input can not be implemented since $F(q,\dot{q})$ in u^* is unknown. This impractical input, u^*, just helps the conversion of (18) to tracking error dynamics. Now, Suppose that the input, u, is the addition of the neural network controller, u_c, and the supervisory controller, u_s, that is

$$u = u_c + u_s \tag{21}$$

Substituting (21) to (18), (18) becomes

$$\ddot{q} = F(q,\dot{q}) + (u_c + u_s). \tag{22}$$

By subtraction and addition of u^*, the equation of error is obtained:

$$\ddot{q}_e = -Ke + (u_c + u_s - u^*) \tag{23}$$

or equivalently in the other vector form,

$$\dot{e} = \Lambda e + B(u_c + u_s - u^*) \tag{24}$$

where

$$\Lambda = \begin{bmatrix} 0 & I \\ -K_1 & -K_2 \end{bmatrix}, B = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{25}$$

Define $V = \frac{1}{2}e^TPe$ where P is a symmetric positive definite matrix satisfying the following Lyapunov equation.

$$\Lambda^T P + P\Lambda = -Q \tag{26}$$

where Q is a positive definite matrix.

Now, the following theorem is derived based on the Lyapunov's direct method.

Theorem 2 *Consider the robot dynamics described by (16), satisfying* Assumption 3 - 4, *and subject to control given by*

$$\tau = \begin{cases} M(q)u_c \; if \; V < V_M \\ M(q)u_s \; if \; V > V_M \end{cases} \tag{27}$$

and

$$\begin{aligned} u_s &\equiv \ddot{q}_d - Ke + u_{s1} & (28) \\ u_{s1} &\equiv -\frac{(e^TPB)^T}{\|e^TPB\|} \\ &\quad \cdot \|M\| \cdot (c_0 + c_1\|\dot{q}\|^2) & (29) \end{aligned}$$

where $V_M > 0$ is a constant specified by the designer. Then, $V < V_M$ as $t \rightarrow \infty$.

Proof when $V > V_M$,

$$\begin{aligned} \dot{V} &= -\frac{1}{2}e^TQe + e^TPB(u - u^*) \\ &= -\frac{1}{2}e^TQe + e^TPB(u_{s1} + F(q,\dot{q})) \\ &\leq -\frac{1}{2}e^TQe + \|e^TPB\|\|F(q,\dot{q})\| \\ &\quad + e^TPBu_{s1} \\ &\leq -\frac{1}{2}e^TQe \\ &\quad + \|e^TPB\|\|M\|(\|N\| - (c_0 + c_1\|\dot{q}\|^2)) \\ &< 0 \end{aligned} \tag{30}$$

□

Identification and Control Procedure: The identification and control procedure of robot dynamics is similar to that of dynamics (1). The neural network identifier plus PD-type controller, u_c, is

$$u_c = -N_F(q,\dot{q}) + \ddot{q}_d - Ke. \tag{31}$$

If N_F is sufficiently accurate to cancel out the function F, robot dynamics becomes asymptotically stable.

$$\dot{e} \cong \Lambda e \tag{32}$$

Even in the case N_F does not approximate F, the supervisory controller guarantees the bounded stability of tracking control.

Simulation : The simple two link robot is considered, which has the following $M(q)$ and $N(q,\dot{q})$.

$$\begin{aligned} M_{11} &= m_1 lg_1^2 + I_1 + m_2(l_1^2 + lg_2^2 \\ &\quad + 2l_1 lg_2\cos(q_2) + I_2 & (33) \\ M_{12} &= m_2(lg_2^2 + l_1 lg_2 cos(q_2)) + I_2 & (34) \\ M_{21} &= M_{12} & (35) \\ M_{22} &= m_2 lg_2^2 + I_2 & (36) \end{aligned}$$

$$\begin{aligned} N_{11} &= -m_2 l_1 lg_2 \sin(q_2)(2\dot{q}_1\dot{q}_2 + \dot{q}_2^2) \\ &\quad + m_1 g lg_1 \cos(q_1) + m_2 g(l_1 cos(q_1) \\ &\quad + lg_2 cos(q_1 + q_2)) & (37) \\ N_{21} &= m_2 l_1 lg_2 \sin(q_2)\dot{q}_1^2 \\ &\quad + m_2 g lg_2 \cos(q_1 + q_2) & (38) \end{aligned}$$

where $m_1 = 2$, $m_2 = 1$, $g = 9.8$, $lg_1 = 0.5$, $lg_2 = 0.5$, $l_1 = 1$, $l_2 = 1$, $I_1 = 0.5$, $I_2 = 0.25$, and q_1,q_2 denote the joint angles.

The desired path in the cartesian coordinate is given by $(x_d(t), y_d(t)) = (0.5 + 0.3\sin(t), 0.5 + 0.3\cos(t))$. q_{d1}, q_{d2} is calculated by inverse kinematics. We choose $V_M = 0.5$, $Q = diag[10,\cdots,10]$, $c_0 = 10$, $c_1 = 10$ and K_1,K_2 are the diagonal matrices whose elements on the diagonal are 2. The neural network which has the structure of $\aleph_{4,10,6,2}$ is used to identify the function $N(q,\dot{q})$. The sigmoid functions for the output nodes are scaled properly according to the variation range of $N_i(q,\dot{q})$, that is,

$$S_o(x) = K\tanh(x) \tag{39}$$

where K is a scale factor.

The following figures show the results of the identification and control of the robot dynamics using neural networks under the supervisory control while learning. Fig.5 depicts the tracking of the circle at the first revolution. Fig.6 shows the tracking of the circle at the 100th revolution. In Fig.6, the tracking is not well done at the upper end of the circle because the sampled data is not as many as the other part of the circle. This confirms our prediction that the uniform samples can not be achieved in the tracking control of robot dynamics.

5 Conclusion and Further Study

This paper is devoted to guarantee the stability in the application of neural network to dynamics system identification. For the certain class of continuous-time systems, supervisory control algorithm is proposed to assure the stability in the sense of boundedness. Trying to get enough samples for neural network training, the system may become unstable. The supervisory controller guarantees the boundedness of the state trajectory. This paper does not deal with the classical neural network problem that what kind of structure is optimal for the given problem. Our efforts are mainly devoted to the problem of stability in getting the samples.

References

[1] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Contr., Signals, Syst.*, vol. 2, no. 4, pp. 303-314, 1989.

[2] T. Chen and H. Chen, "Approximations of continuous functionals by neural networks with application t o dynamic systems," *IEEE Trans. Neural Network*, vol. 4, no. 6, pp. 910-918, 1993.

[3] N. E. Cotter, "The Stone-Weierstrass theorem and its application to neural networks," *IEEE Trans. Neural Network*, vol. 1, no. 4, pp. 290-295, 1990.

[4] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4-27, 1990.

[5] A. U. Levin and K. S. Narendra, "Control of nonlinear dynamical systems using neural networks: controllability and stabilization," *IEEE Trans. Neural Networks*, vol. 4, no. 2, pp. 192-206, 1993.

[6] K. S. Narendra and A. U. Levin, "Regulation of nonlinear dynamical systems using multiple neural networks," *Proc. Amer. Contr. Conf.*, pp. 1609-1614, 1991.

[7] L. Wang, "Stable adaptive fuzzy control of nonlinear systems," *IEEE Trans. Fuzzy Systems*, vol. 1, no. 2, pp. 146-155, 1993.

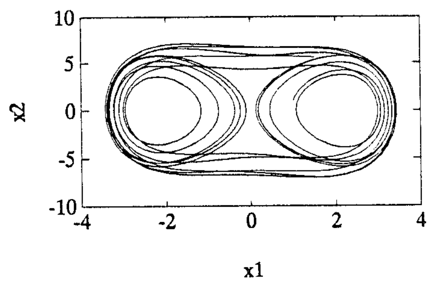

Fig.1 Trajectory of chaotic system (16) in the phase plane with $u(t) = 0$ and $x_1(0) = x_2(0) = 2$.

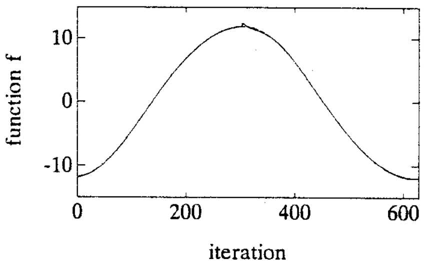

Fig.4 Approximation of $f(\mathbf{x}, t)$ by $N_f(\mathbf{x}, t)$ at the 1000th revolution.(- -: $f(\mathbf{x}, t)$, —: $N_f(\mathbf{x}, t)$)

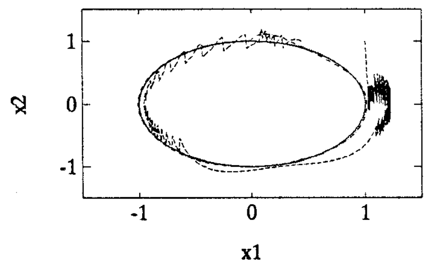

Fig.2 Tracking control when the states revolves 1, 200, 1000 times around the unit circle.(—: desired trajectory, - -: 1th revolution, -·-: 100th revolution, · · ·: 1000th revolution)

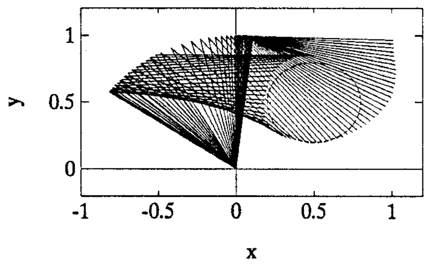

Fig.5 Tracking control at the first revolution.(- -: desired trajectory)

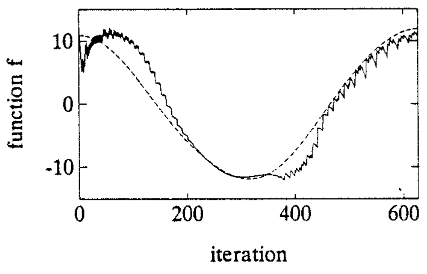

Fig.3 Approximation of $f(\mathbf{x}, t)$ by $N_f(\mathbf{x}, t)$ at the first revolution.(- -: $f(\mathbf{x})$, —: $N_f(\mathbf{x})$)

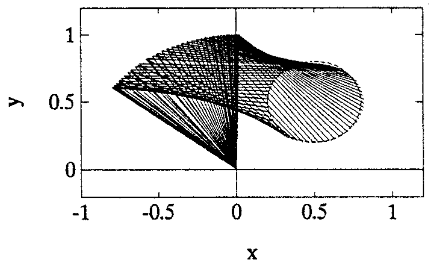

Fig.6 Tracking control at the 100th revolution.(- -: desired trajectory)