# Intelligent Sliding Mode Control in Flexible Structures

Dezhi Li

A thesis presented to Lakehead University

In partial fulfillment of the requirement for the degree of

Master of Science in Control Engineering

Thunder Bay, Ontario, Canada, 2010

# Abstract

The first objective of this work is to develop an intelligent sliding mode controller for vibration control in flexible structures. The proposed control consists of two processes: system identification and sliding mode control. System identification is performed based on a neural fuzzy (NF) approximator. A novel extended gradient method and a modified least square estimate (LSE) algorithm are proposed for neuro-fuzzy system training. The training is performed in a hybrid approach: the nonlinear parameters in the NF approximator are updated using the extended gradient method while the linear parameters are optimized by the modified LSE. In system control, an enhanced sliding mode (ESM) control system is developed to promote the control effort for active vibration suppression especially in flexible structures. Based on experimental investigation, when the principle of the terminal attractor is used in the classical gradient descent algorithm or sliding mode control systems, it causes implementation problems because the initial condition should be nonzero. The proposed training techniques provide faster convergence while avoiding the associated implementation problems. The stability of the proposed training techniques is demonstrated by the Lyapunov analysis. The effectiveness of the developed techniques is verified experimentally with a flexible structure experimental setup. Test results show that the suggested hybrid training technique can effectively improve the convergence of the NF approximator; the ESM controller can efficiently perform vibration suppression in flexible structures and easy to implement.

The commonly used global search method is genetic algorithm (GA). The problems in the classical GA are low convergence speed and lack of fast global search capability for complex search space. The second objective of this work is to develop a more efficient global training

approach, called enhanced genetic algorithm (EGA) for system training and optimization applications. Two approaches are proposed: Firstly, a novel group-based branch crossover operator is suggested to thoroughly explore local space and speed up convergence. Secondly, an enhanced MPT (Makinen-Periaux-Toivanen) mutation operator is proposed to promote global search capability for complex search space. The effectiveness of the developed EGA is verified by simulations based on a series of benchmark test problems. Test results show that the branch crossover operator and enhanced MPT mutation operator can effectively improve the convergence speed and global search capability. The EGA technique outperforms other related GA methods with respect to global search efficiency and operation efficiency.

# Acknowledgements

I wish to express my sincere appreciation to Dr. Wilson Wang. His guidance and support were essential to the completion of this thesis. I would like to thank my co-supervisor Dr. B. Ismail for his help and suggestions. I would also like to acknowledge Dr. Abdelhamid Tayebi and Dr. Rachid Benlamri for their efforts as examiners of this thesis.

I thank all my friends at Lakehead University for their friendship, support and technical discussion during the past two years.

Most importantly, I want to thank my parents for their encouragement and support on my graduate studies.

# Contents

# List of figures

# List of Tables

# Chapter 1 Introduction

## 1.1 Literature review

Flexible structures are widely used in various engineering applications such as robots, buildings and vehicles, because of their special attributes such as light weight, compactness and easy implementation. The flexible structures contain infinite number of vibration modes, and have low rigidity and small material damping. Even a small external excitation may lead to large amplitude vibration and/or long vibration damping time. Vibration suppression in flexible structures has attracted increasing attentions in R&D in recent years and is critically needed in many engineering applications [1-4]. There are many techniques dealing with nonlinear system control; for example, static and dynamic output feedback control [5], fuzzy control [6] and periodic control algorithms [7]. Among them, sliding mode (SM) control is one of the widely employed techniques for the vibration control of flexible structures [8-15], which performs discontinuous control actions in order to force the system states to achieve the defined sliding surface and maintain on it. SM control is usually fast in error convergence and insensitive to bounded input disturbances, nonlinearities and uncertain dynamics [16]. Several SM-based techniques have been proposed in literature for vibration control; for example, SM plus fuzzy control [17], SM with command input shaping control [18], a multirate output feedback based discrete SM control [19] and robust SM control [20-22]. However, the error of linear sliding surface will not converge to zero in finite time and the convergence of linear sliding surface is relatively slow. Recently, a novel SM control termed as the terminal sliding mode (TSM) control algorithm has been studied to further enhance control performance [23-27]. Compared with linear hyper-plane based sliding modes, TSM offers some superior properties such as fast and

finite time convergence. Such a controller is useful for high precision control, such as the second or high nonlinear uncertain systems control, because it can speed up the rate of convergence near the equilibrium point.

On the other hand, the SM control and the TSM control have high-precision control with asymptotic stability; however, they may cause slow convergence when the error is close to origin or far away from origin, respectively [18]. Fast TSM integrates the merits of SM control and TSM control so as to achieve fast transient convergence both at a distance from and at a close range of equilibrium [19]. Although the terminal attractor can provide the exponentially growing rate of convergence when the state is near the equilibrium [19], the initial condition should not be set zero, because large derivative of terminal attractor will lead to a large control signal when state is close to origin, which may result in failure in real implementation. This type of flaw makes it unable to implement these controllers in many real applications because certain problems arises such as the control signal becomes too large or even out of limitation when the initial condition is zero.

During SM control implementation, undesirable phenomenon of oscillations may occur with finite frequency and amplitude, which is known as ‘chattering’. Chattering is a harmful phenomenon because it leads to low control accuracy, high wear of moving mechanical parts, and high heat losses in electrical power circuits [61]. Thus it becomes one of the main problems in implementation of SM control. This time we use intelligent tools to perform system identification as an alternative to force system state approach and maintain on the sliding surface instead of sign function.

Computational intelligence has provided more flexible tools in system control. Fuzzy logic

(FL) simulates human reasoning and employs a set of IF-THEN rules to model complex processes [37]. FL has been used for vibration suppression in flexible beams [28-30]; for example, FL control of the end-point vibration in an experimental flexible beam [31], T-S fuzzy bilinear model and fuzzy controller designed for a class of nonlinear systems [32], and T-S fuzzy model used for time-delay chaotic systems control [33]. FL control has been proven effective for complex, nonlinear and imprecisely defined systems for which classical model-based control techniques are impractical or even impossible. However, in FL control design, a common bottleneck is how to properly set up fuzzy control rules, which are usually time-consuming and difficult. In general, fuzzy rules are established by expertise; however, human knowledge may not be transcribed properly into the requisite rule form, especially for systems involving many inputs. Fuzzy systems lack self-tuning capabilities to improve control performance based on different system conditions. The related fuzzy rules and membership functions (MFs) for the input/output variables have to be optimized by trial and error. Unfortunately there exists no formal framework for the choice of the fuzzy sets and parameters of fuzzy systems.

The neural networks (NNs) use a set of interconnected nodes to approximate any arbitrary continuous function. NNs can be trained to adjust its inter-connections among nodes to achieve optimal or near optimal input-output mappings and to accommodate even poorly modeled and nonlinear dynamical systems. NNs possess the tolerant capability in reasoning operations under noisy environments, for example, with faulty and missing data. NNs-based control systems have been proposed in literature for vibration suppression in flexible beams [34, 35]. NNs-based controllers, however, may not be suitable for some linear or linearizable systems control, which can result in degradation in performance in terms of computation time and controller convergence [36]. Furthermore, the reasoning in NNs is opaque to users and the extracted

distributed knowledge after training is usually difficult to understand by designers.

A neuro-fuzzy (NF) paradigm is a synergetic system of FL and NNs, in which FL provides the linguistic control reasoning structure whereas the control system (i.e., structure and parameters) is updated by NNs-based training algorithms [37]. It aims at integrating the strengths of both FL and NNs. Functionally, the NF system performs as a function approximator. Theoretical investigations have shown that an NF system can approximate any continuous function to any accuracy with a bounded error tolerance when sufficient fuzzy rules and appropriate fuzzy sets are employed [38].

Once the control reasoning scheme is established, the control system parameters should be trained properly to improve the control performance. There are several learning algorithms available in literature for training an NF scheme such as gradient descent (GD) algorithm [39], least square estimate (LSE) [40], momentum gradient [41], exponential gradient [42], etc. Recently the fast TSM is incorporated into the GD to improve the control convergence [43]; when that training technique is applied on systems with initial error to be zero or in close range of origin, it will generate problems such as no sufficient small sampling time and/or proper tolerance for system implementation.

In general, the aforementioned training methods belong to the category of local space search. The most commonly used global search method is GA. GA is an adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection [37]. It explores a random search within a defined search space to achieve a minima, which has been used in many optimization [44-46] and training [47-49] applications. GA is a derivative-free stochastic optimization algorithm which has several advantages over other related classical training

methods (e.g., GD and LSE): 1) it can be used for parallel-processing operations, 2) its search space (both continuous and discrete) is more flexible, and 3) its stochastic attribute can help to escape some of the local minima [37]. Different from the classical optimization algorithms, GA operates on a set of points as a population which is then evolved toward a higher fitness grade [50]. In each generation, the GA constructs a new population using some genetic operators such as the elitism, crossover and mutation. Individuals with higher fitness are more likely to be selected to participate in mating operations. The entire population is upgraded to form a new generation with a higher fitness grade. Usually, individuals are denoted as binary strings. In general applications, however, mapping real values to binary strings may degrade processing efficiency [51] and precision [52]. To solve these problems, several operators have been designed in real coded GA-based optimizations. For example, the Laplace crossover was suggested in [53] with the Makinen-Periaux-Toivanen (MPT) mutation to speed up search operations. A parent-centric real-parameter crossover operator was proposed in [54] to select male and female parents separately with the offspring close to the female parent.

On the other hand, several approaches have been proposed in the literature to improve the convergence of GA, for example, based on nonlinear transformation of genetic operators [55], a feedback dynamic penalty function [56], an annealing function [57], and a differential operator [58]. However, GA with these solutions still has some apparent shortcomings such as slow convergence (time-consuming) and relatively poor capability to attain global minima when the search space is complex.

## 1.2 Objectives of Research

To tackle the aforementioned challenges, the first objective of this work is to develop an intelligent sliding mode control, or ESM technique, to improve the performance of SM control and overcome the proposed problems of fast TSM control. The ESM controller should be easy in implementation while retain the merit of fast convergence of the fast TSM Control. System performance should be further improved by new and stable training techniques.

Another objective of this work is to develop a novel global training technique to improve the global convergence, training efficiency and accuracy. The strategy is to use a novel crossover operator and a mutation operator to speed up the search process, expand the local search space, and enhance global search capability.

## 1.3 Contributions of this work

1) An extended sliding mode control technique has been developed to provide better vibration control in flexible structures [62]. It is proposed to solve the related implementation problems in the classical fast TSM control while retains its fast convergence merit. Its superior performance has been verified by experiments.

2) An enhanced gradient method has been proposed to update the NF system where the proposed new item is embedded in conventional gradient algorithm [62]. It brings improved convergence speed both nearby and far away from the origin and overcome the proposed implementation problem. The Lyapunov stability of the proposed EG method is proven.

3) A modified LSE technique has been suggested to update the NF system. It brings improved convergence speed both nearby and far away from the origin which is superior to classic LSE algorithm. The Lyapunov stability of the proposed modified LSE is proven.

4) An enhanced GA method has been developed to promote training efficiency and accuracy [63]. A new branch crossover (BC) operator is proposed to speed up the search process and expand the local search space. On the other hand, an enhanced Makinen-Periaux-Toivanen mutation (EMM) operator is developed to enhance global search capability of the EGA technique.

The effectiveness of all the proposed techniques has been verified by tests and/or simulations in Chapter 5 and Chapter 7.

## 1.4 Thesis Organization

After the Introduction as described in this chapter, the remainder of this thesis is organized as follows:

Chapter 2 describes the theoretical background of the related computational intelligence tools such as fuzzy logic, neural networks and neuro-fuzzy schemes.

In Chapter 3, the limitations of the classical GD and LSE algorithms are discussed first; and then developed enhanced gradient method and modified LSE are systematically discussed. The parameters in the NF approximator are fine-tuned using a hybrid training technique based on these two new methods.

In Chapter 4, the experimental workstation used in this work is described first; and then the proposed the ESM control is discussed in details.

The effectiveness of the developed ESM control and two novel training techniques are verified experimentally in Chapter 5. A pulse disturbance is employed to excite the resonance of the smart structure. Some extra blocks are attached at different places of the flexible structure to test the robustness of the controller.

The developed EGA technique is discussed in Chapter 6. GA technique is reviewed first. Afterwards the suggested branch crossover operator and EMM method are discussed.

The effectiveness of the EGA techniques is verified by simulation in Chapter 7. Twenty benchmark problems are used to prove the advantages of proposed branch operator and EMM method.

Finally, some conclusion remarks and future research projects are summarized in Chapter 8.

# Chapter 2 Theoretical Background in Intelligent Systems

Real-world problems require intelligent systems that combine knowledge, techniques and methodologies from various sources. These intelligent systems are supposed to possess human-like expertise within a specific domain, adapt themselves in changing environments, and explain how to make decisions or take actions. NNs recognize patterns and adapt themselves to cope with changing environments; FL incorporates human knowledge and performs inference and decision making. The integration of these two approaches results in neuro-fuzzy (NF) system. In confronting real-world computing problems, appropriate integration of more than one computing technique could result in a synergetic system which can overcome some shortcomings of the component techniques.

For the convenience of readers, the theoretical background of the related intelligent computational techniques such as FL, NNs, and NF schemes is briefly described in this chapter.

## 2.1 Fuzzy Logic Models

The classical reasoning sets have crisp boundaries. Although these classical sets are suitable for mathematics and computer operations, they do not reflect the nature of human reasoning which tend to be abstract and imprecise. Fuzzy logic (FL) employs a set of linguistic IF-THEN rules to mimic human reasoning in comparable circumstances. Without a crisp boundary, fuzzy sets use smooth transition membership functions (MFs) in system modeling. Consider an example as follows:

*R1*: IF (Service is *good*) AND (Food is *delicious*), THEN (Tips are *very generous*),

*R2*: IF (Service is *good*) AND (Food is *normal*), THEN (Tips are *generous*),

*R3*: IF (Service is *normal*) AND (Food is *delicious*), THEN (Tips are *normal*),

⋮

*R9*: IF (Service is *bad*) AND (Food is *bad*), THEN (Tips are *mean*),

where each input "Service" or "Food" has three MFs, namely, (good, normal and bad) and (delicious, normal and bad). *Ri* is the fuzzy rule ($i$ =1, 2… 9). "Tips" is the output of this inference system.

A number of fuzzy inference systems have been suggested in the literature, such as Mamdani fuzzy models (also known as linguistic fuzzy systems) and the Takagi-Sugeno (TS) fuzzy systems. In Mamdani fuzzy models, fuzzy rule (e.g. "Tips are *very generous*") is defined as a consequent fuzzy set. If fuzzy rule is a polynomial of the input variables ("Service" and "Food"), it is called the first order TS model, or TS1. Otherwise, if fuzzy rule is a constant, it becomes the zeroth order TS fuzzy model**,** or TS0. TS0 can be viewed as a special case of the Mamdani fuzzy inference system, in which the consequent part of each rule is a fuzzy singleton. The three types are shown schematically based on the aforementioned examples:

1) Type I (Mamdani model): Fuzzy model output is a consequent fuzzy set (e.g. Tips are *generous* or *normal*);

2) Type II (Zeroth-order Takagi-Sugeno (TS) model): Fuzzy model output is a singleton (e.g. Tips are *5 dollar*)

3) Type III (First-order TS model): Fuzzy model output is a linear function (e.g. Tips are calculated as $0.5\times x+0.5\times y$) where $x$ is "Service" and $y$ represents "Food".

## 2.2 Neural Networks

NNs consist of a group of inter-connected nodes to map the input space to the output space for different reasoning operations. A node is an information-processing unit as shown in Figure 2.1.

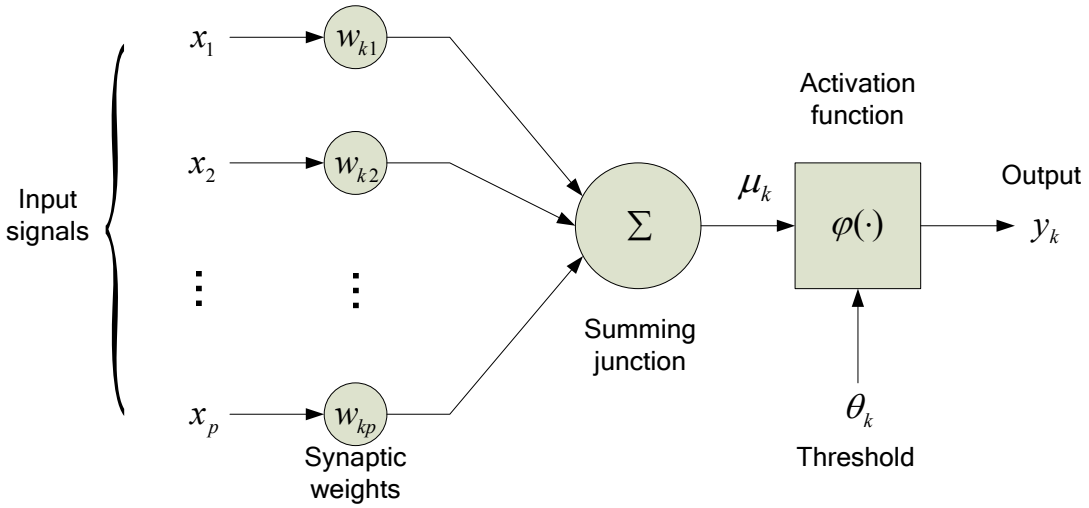


Figure 2.1. Nonlinear model of a node.

Mathematically, the function of a node *k* can be described by the following equations:

$$\mu_k = \sum_{j=1}^{p} w_{kj} x_j \tag{2-1}$$

and $y_k = \varphi(\mu_k - \theta_k)$ (2-2)

where $x_1, x_2, \ldots, x_p$ are the input signals; $w_{k1}, w_{k2}, \ldots, w_{kp}$ are the synaptic weights of node *k*; $\mu_k$ is the linear combined output; $\theta_k$ is the threshold; $\varphi(\cdot)$ is the activation function; and $y_k$ is the output of the node.

Consider NNs as shown in Figure 2.2, which contains three inputs, two outputs, and three hidden nodes. For simplicity, each node in this network uses the same activation function. For instance, the activation function of node 7 is

$$x_7 = \frac{1}{1+e^{-(w_{4,7}x_4+w_{5,7}x_5+w_{6,7}x_6)}} \tag{2-3}$$

where $x_4$, $x_5$, and $x_6$ are outputs from nodes 4, 5, and 6, respectively, and the parameter set of node 7 is denoted by $\{w_{4,7}, w_{5,7}, w_{6,7}\}$. $w_{i,j}$ is the weight associated with the link connecting node $i$ and $j$.

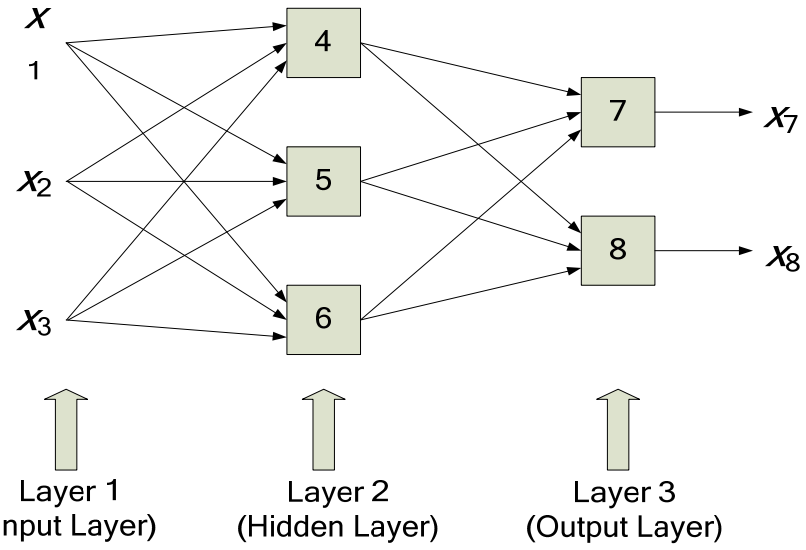


Figure 2.2. NNs model.

NNs should be trained so as to achieve an optimal input-output mapping. The type of learning is determined by the manner in which the parameter updates take place. One of the main properties of NNs is its ability of learning from its environment so as to improve its

performance. After the network is properly trained, it will become more knowledgeable about its environment.

## 2.3 Neuro-Fuzzy Systems

As discussed in Chapter 1, both FL and NNs have their own advantages and disadvantages. An NF system is a synergetic scheme of FL and NNs, in which the FL provides the reasoning architecture and NNs contribute to the training methods. The schematic representation of NF is given in Figure 2.3.

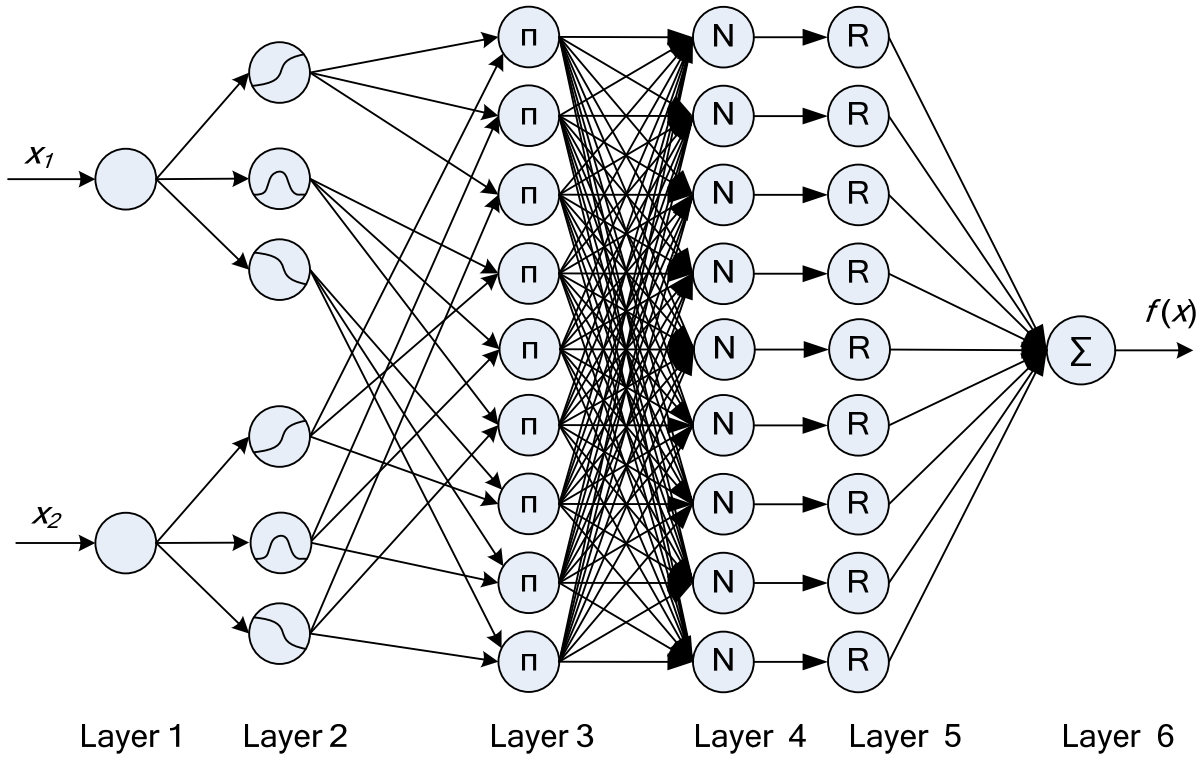


Figure 2.3 NF network architecture.

Layer 1: This layer contains two nodes which are two inputs of the NF system.

Layer 2: Every node in this layer is an adaptive node with membership function. The membership function represents the membership grade of a fuzzy set and it specifies the degree to which the given input satisfies the quantifier. Parameters in this layer are referred to as premise parameters.

Layer 3: Every node in this layer is a fixed node labelled π whose output is the product of all the incoming signals. Each node output represents the firing strength of a fuzzy rule.

Layer 4: Every node in this layer is a fixed node labelled N. The node in this layer normalize the input firing strength. For convenience, outputs of this layer are called normalized firing strengths.

Layer 5: The single node in this layer is a fixed node labelled R, which computes the multiplication of input normalized firing strength and the corresponding fuzzy rule.

Layer 6: The single node in this layer is a fixed node labelled Σ, which computes the overall output as the summation of all incoming signals.

Generally NF system is able to approximate a nonlinear function with a small bounded residual error [38]. Thereafter it is usually employed for system identification as will be discussed in Chapter 3.

## 2.4 System Training

The parameters in the NNs and NF schemes should be trained properly to optimize input/output space mapping. Many training techniques have been proposed in literature, which can be categorized into two classes: derivative-based training techniques and derivative-free training techniques. Derivative-based techniques are capable of determining search direction

according to an objective function's derivative information. Gradient descent method and Newton's method form the foundation of many gradient-based algorithms. Actually, many algorithms can be regarded as a form of compromise between gradient descent and Newton's methods. A class of the gradient-based methods can be applied to optimizing nonlinear NF models. In fact, steepest descent and conjugate gradient methods are major algorithms used for NNs learning.

The LSE is another widely employed algorithm because the sum of squared errors is chosen as the objective function to be minimized in many cases. Those methods are commonly used in data fitting and regression involving nonlinear models. Heuristically informed search techniques are employed in many artificial intelligence applications. When a search space is too large for an exhaustive search and global minima is required when dealing with complex search spaces, the GA would be a candidate technique for population-based systematic random searches.

In system training, the gradient method is one of the classical techniques for parameter optimization over multidimensional input/output spaces. It is the basic algorithm for many advanced training methods for both constrained and unconstrained problems. Moreover, despite its slow convergence, the gradient method is still a common algorithm used for nonlinear optimization due to its simplicity. Least square estimate (LSE) is an alternative method for linear parameter training in NF models. By decreasing the square error, LSE adjusts the parameters to achieve the minimization of objective function. The related properties of these classical techniques will be discussed in Chapter 3 for advanced analysis.

# Chapter 3 Training Techniques for NF System Optimization

The NF approximator as well as the developed extended gradient method and modified LSE training techniques will be discussed in this chapter.

## 3.1 The Neuro-Fuzzy Approximator

An NF approximator is proposed first in this chapter to compensate the nonlinear functions. It is a four-layer NF scheme as illustrated in Figure 3.1. Layers 1 and 2 constitute the antecedent part, and layers 3 and 4 correspond to the consequent part. The links have unity weights unless specified.

The network architecture is schematically shown in Figure 3.1:

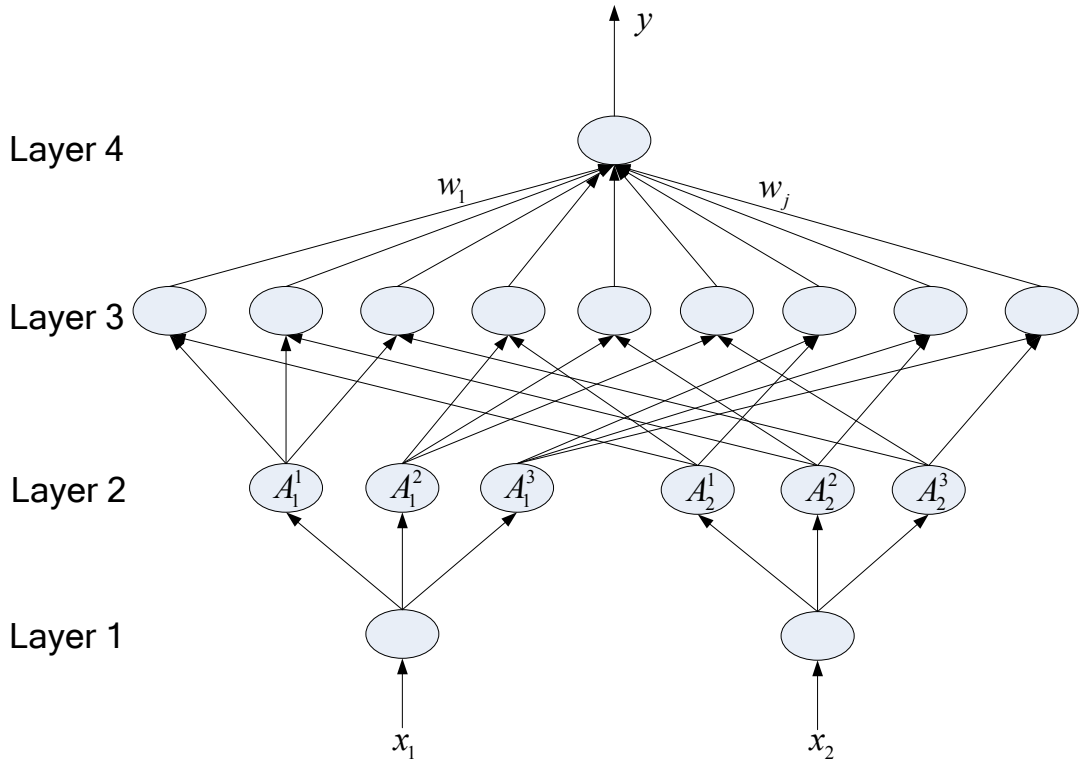


Figure 3.1. The network architecture of the NF approximator.

- Layer 1 is the input layer: The input variables $x_1$ and $x_2$ are fed to the network; the nodes in this layer just transmit input values to the next layer.

$$O_i^1 = x_i \tag{3-1}$$

- Layer 2 is for fuzzification: The input variables are fuzzified in this layer. If Gaussian and Sigmoid membership functions are applied in this case, the node outputs with respect to each input node will be

$$O_i^1 = S(x_i, a_i^1, c_i^1) = \frac{1}{1+\exp(-a_i^1 \cdot (x_i - c_i^1))}; \tag{3-2}$$

$$O_i^2 = G(x_i, a_i^2, c_i^2) = \exp(-\frac{1}{2} \cdot (\frac{x_i - c_i}{a_i})^2) ; \quad (3\text{-}3)$$

$$O_i^3 = S(x_i, a_i^3, c_i^3) = \frac{1}{1 + \exp(-a_i^3 \cdot (x_i - c_i^3))} ; \quad (3\text{-}4)$$

where $a_i^j$ and $c_i^j$ represent the center and spread of the Gaussian membership function in Equation (3-3). They also denote the center and slope of the sigmoid membership function respectively in Equations (3-2) and (3-4). In $a_i^j$ and $c_i^j$, $i$ represents the number of input variables and $j$ corresponds to number of membership functions with respect to each input variable.

- Fuzzy operation is taken in Layer 3: The links in this layer perform precondition matching of fuzzy rules. Every node performs a specific fuzzy operation. If a product operator is used in this case, the node output becomes

$$O^v = O^i \cdot O^j \quad (3\text{-}5)$$

where the upper subscript $v$(=1,2,…,9) is the corresponding number of the fuzzy rules in layer 3. $i$(=1,2,3) and $j$(=1,2,3) are respective number of membership functions corresponding to input variables $x_1$ and $x_2$.

- Defuzzification is performed in Layer 4. If a centroid defuzzification operator is used, after normalization of the input signals from the previous layer, the system output becomes,

$$y = \frac{\sum_{v=1}^{M} O^v \cdot w_v}{\sum_{v=1}^{M} O^v}, \quad (3\text{-}6)$$

where $y$ is the output variable, $w_v$ is the weight of link between layer 3 and 4.

The NF system parameters will be optimized using the proposed training techniques. In this case, an extended gradient (EG) method and a modified LSE algorithm are proposed to train the nonlinear and linear system parameters, respectively, which will be discussed as follows.

## 3.2 The Extended Gradient Training Method

### 3.2.1 The Classical Gradient Algorithm

As stated before, the gradient algorithm is one of the classical techniques for the optimization of nonlinear parameters in both constrained and unconstrained problems. The feasible descent directions can be determined by deflecting the gradients through multiplication by $\boldsymbol{G}$ (i.e., deflected gradients):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \boldsymbol{G}\boldsymbol{g} \tag{3-7}$$

where $\boldsymbol{w}_t$ and $\boldsymbol{w}_{t+1}$ are the trained parameters at time step $t$ and $t+1$ respectively. Equation (3-7) also contains positive learning rate $\eta$ and positive definite matrix $\boldsymbol{G}$.

The gradient descent method is a fundamental method in this category with $\boldsymbol{G}=\boldsymbol{I}$ shown in Figure 3.2. Most revised gradient methods (e.g., Newton's method and the Levenberg-Marquardt method) possess the form of Equation (3-7) to bias the negative gradient direction ($-\boldsymbol{g}$) by choosing different representation of matrix $\boldsymbol{G}$.

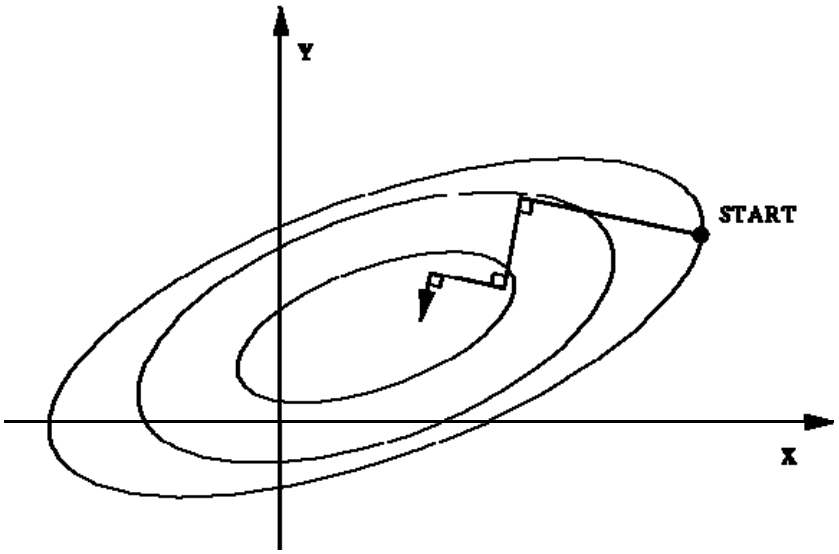


Figure 3.2. Gradient descent method approaches the minima in a zig-zag manner.

Ideally, we wish to find a value of $\boldsymbol{w}_{t+1}$ that satisfies the following:

$$g(\boldsymbol{w}_{t+1}) = \frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}}\Big|_{\boldsymbol{w}=\boldsymbol{w}_{t+1}} = 0 \tag{3-8}$$

In practice, however, it is difficult to solve Equation (3-8) analytically. To minimize the objective function, the descent procedures are usually repeated until one of the following stopping criteria is satisfied:

1) The objective function value is sufficiently small;

2) The length of the gradient vector $\boldsymbol{g}$ is smaller than a specified value; or

3) The specified computing time is exceeded.

Moreover, despite its slow convergence, the method is the most frequently used nonlinear optimization technique due to its simplicity.

### 3.2.2 Extended Gradient (EG) Method

In system training, the classical gradient method is very slow in convergence especially when the approximation error becomes small. To tackle this problem, an EG method is proposed in this subsection.

According to the universal approximation property [38], any continuous function can be approximated by an NF paradigm to an expected accuracy with a bounded error:

$$y = \boldsymbol{w}^T \xi(\boldsymbol{w}) \tag{3-9}$$

$$y_d = \boldsymbol{w}^{*T} \xi(\boldsymbol{w}) - \varepsilon(\boldsymbol{w}) \tag{3-10}$$

where $y$ denotes the output of the NF approximator and $y_d$ is the desired value of $y$. $\boldsymbol{w} = [w_1, w_2, \cdots\cdots w_M]^T$ is a nonlinear weight vector in the NF approximator, in which the components are variables that need to be adjusted; $\boldsymbol{w}^* = [w_1^*, w_2^*, \cdots\cdots w_M^*]^T$ is the desired weight vector, in which all entries are optimal nonlinear weight values (constants). $M$ is the size of weights vector $\boldsymbol{w}$. $\xi(\boldsymbol{w}) = \dfrac{\partial y}{\partial \boldsymbol{w}}$. $|\varepsilon(\boldsymbol{w})| \le \varepsilon_M$, where $\varepsilon(\boldsymbol{w})$ is the optimal approximation error tolerance, and $\varepsilon_M$ is a small positive constant. It should be noted that $\varepsilon(\boldsymbol{w})$ is a bounded unknown function of system states, which cannot be estimated directly by the adaptive law as discussed in [43].

The approximation error $e$ can be expressed as

$$e = y - y_d = (\boldsymbol{w} - \boldsymbol{w}^*)^T \xi(\boldsymbol{w}) + \varepsilon(\boldsymbol{w})$$

$$= \tilde{\boldsymbol{w}}^T \xi(\boldsymbol{w}) + \varepsilon(\boldsymbol{w}) \tag{3-11}$$

where $\tilde{\boldsymbol{w}} = \boldsymbol{w} - \boldsymbol{w}^*$ is the estimation error of the optimal weight $\boldsymbol{w}^*$. In this work, the proposed EG method is to speed up convergence. The cost function of the EG is established as

$$E(t) = \frac{1}{2} e(t)^2 + \frac{\eta_2}{\eta_1} \int_{t_1}^{t_2} T(|e(t)|) dt \,. \tag{3-12}$$

where $\eta_1$ and $\eta_2$ are learning rates and $t_1$ and $t_2$ are starting time point and end time point of each update process. *T(.)* is shown as follows:

$$T(x) = \begin{cases} \dfrac{1}{1+\exp[-\omega(x-\upsilon)]} & if\ x > 0 \\ 0 & if\ x = 0 \\ -\dfrac{1}{1+\exp[\omega(x+\upsilon)]} & if\ x < 0 \end{cases} \tag{3-13}$$

Different from the classical gradient algorithm $\dot{\boldsymbol{w}} = (-\eta_1 e)\xi(\boldsymbol{w})$, the update law in the proposed EG method is

$$\begin{aligned} \dot{\boldsymbol{w}} &= -\eta_1 \frac{\partial E}{\partial y} \frac{\partial y}{\partial \boldsymbol{w}} \\ &= -\eta_1 (e + \frac{\eta_2}{\eta_1} T(e)) \xi(\boldsymbol{w}) \\ &= (-\eta_1 e - \eta_2 T(e)) \xi(\boldsymbol{w}) \end{aligned} \tag{3-14}$$

Because the optimal weight vector $\boldsymbol{w}^*$ does not vary with time, we consider the following Jacobian of classical gradient algorithm around the minimum $\tilde{\boldsymbol{w}} = 0$,

$$J = \frac{\partial \dot{\tilde{w}}_l}{\partial \tilde{w}_l} = \frac{\partial \dot{w}_l}{\partial \tilde{w}_l}$$

$$= \frac{\partial}{\partial \tilde{w}_l}(-\eta_1 e \xi_l(\boldsymbol{w}))$$

$$= \frac{\partial}{\partial \tilde{w}_l}(-\eta_1 \tilde{w}_l \xi_l(\boldsymbol{w}) \xi_l(\boldsymbol{w}))$$

$$= -\eta_1 \xi_l^2(\boldsymbol{w}) \tag{3-15}$$

where $l = 1, 2, \cdots, M$.

For scalar $\tilde{w}_l$, $J$ is the eigenvalue of the first-order approximation matrix. Therefore, the approximation process assumes an exponential convergence property: the convergence rate is very fast when the NF approximator is far away from the minimum $\tilde{w}_l = 0$, but very slow when the approximator is close to $\tilde{w}_l = 0$. That is, the closer the state to the minimum, the slower the rate converges to $\tilde{w}_l = 0$ [43].

In addition, although a terminal attractor can be employed in gradient algorithm to accelerate convergence [43], it will generate implementation problems for the discrete-time control; the sampling time cannot be small enough to evaluate very large acceleration in real applications. To tackle this problem, a $T(.)$ function is introduced in this work to bound the acceleration and to speed up convergence by tuning parameters $\omega$ and $\upsilon$, especially when error approaches zero.

Consider the Jacobian around the minimum $\tilde{w}_l = 0$ in the EG method,

$$J = \frac{\partial \dot{\widetilde{w}}_l}{\partial \widetilde{w}_l} = \frac{\partial \dot{w}_l}{\partial \widetilde{w}_l}$$

$$= \frac{\partial}{\partial \widetilde{w}_l}((-\eta_1 e - \eta_2 T(e))\xi_l(\boldsymbol{w}))$$

$$= \frac{\partial}{\partial \widetilde{w}_l}((-\eta_1 \widetilde{w}_l \xi_l(\boldsymbol{w}) - \eta_2 \widetilde{w}_l \frac{dT(e)}{de}\xi_l(\boldsymbol{w}))\xi_l(\boldsymbol{w}))$$

$$= (-\eta_1 - \eta_2 \frac{dT(e)}{de})\xi_l(\boldsymbol{w})^2 \qquad (3\text{-}16)$$

Compared Equations (3-16) with (3-15), the update rate of $\widetilde{w}_l$ in EG is larger than that of gradient algorithm. Thus *T(.)* is capable of speeding up the convergence in training even when the error is small. Figure 3.3(a) shows an example of *T(.)* when $\omega$=15 and $\upsilon$=0, whereas the convergence acceleration of *T(.)* is depicted in Figure 3.3(b). It can be seen that the value of *y=T(.)* function is larger than *y=x* as states approach the equilibrium over the range $x \in [-0.2, 0.2]$ which implies *T(.)* can improve the convergence in training techniques.

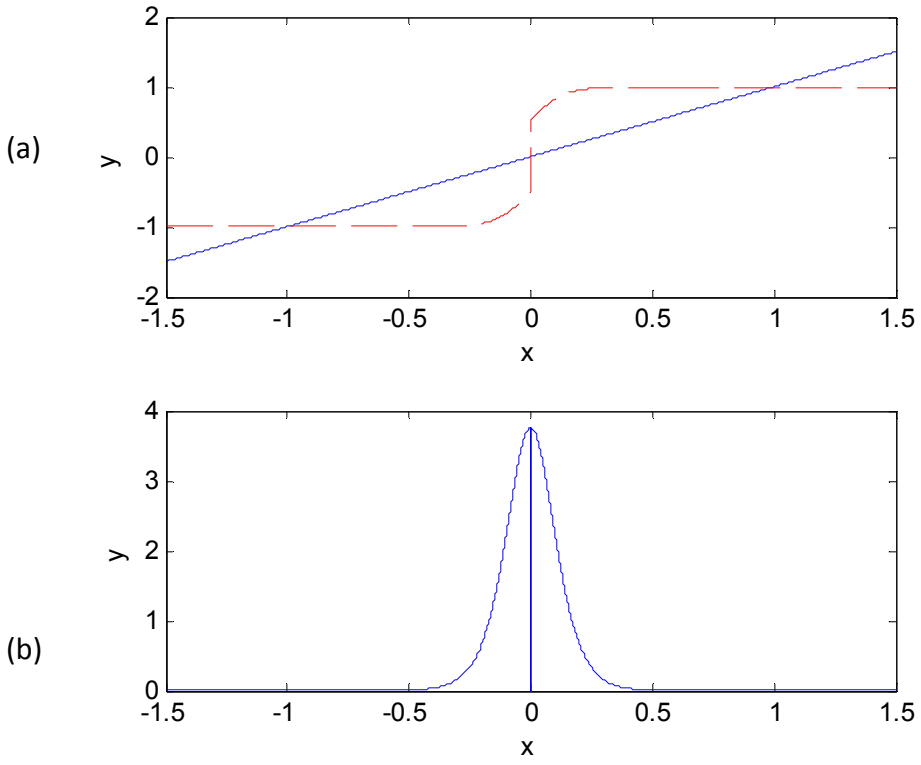


Figure 3.3. (a) The graph of $y=T(x)$ (dashed line) and $y=x$ (solid line). (b) The graph of function $y=dT(x)/dx$.

### 3.2.3 Stability Analysis of the EG method

Firstly, it is easy to prove $xT(x) \geq 0$. Since $\boldsymbol{w}^*$ is optimal weight vector, $\dot{\boldsymbol{w}}^*$ is zero. Given Eq. (3-14) we have

$$\begin{aligned} \dot{\tilde{\boldsymbol{w}}} &= \dot{\boldsymbol{w}} - \dot{\boldsymbol{w}}^* \\ &= \dot{\boldsymbol{w}} \\ &= (-\eta_1 e - \eta_2 T(e))\xi(\boldsymbol{w}) \end{aligned} \tag{3-17}$$

The Lyapunov function and its derivative are defined as

$$V = \frac{1}{2}\tilde{\boldsymbol{w}}^{T}\tilde{\boldsymbol{w}} \tag{3-18}$$

$$\begin{aligned}\dot{V} &= \tilde{\boldsymbol{w}}^{T}\ \dot{\tilde{\boldsymbol{w}}} \\ &= (-\eta_1 e - \eta_2 T(e))\tilde{\boldsymbol{w}}^{T}\xi(\boldsymbol{w}) \\ &= (-\eta_1 e - \eta_2 T(e))(e - \varepsilon(\boldsymbol{w})) \end{aligned} \tag{3-19}$$

If $e > 0$,

$$\dot{V} = (-\eta_1 e - \eta_2 \frac{1}{1+\exp[-\omega(e-\upsilon)]})(e - \varepsilon(\boldsymbol{w}))$$

$$= -\eta_1 |e|^2 - \eta_2 \left|\frac{e}{1+\exp[-\omega(e-\upsilon)]}\right| + \eta_1 e\varepsilon(\boldsymbol{w}) + \eta_2 \frac{\varepsilon(\boldsymbol{w})}{1+\exp[-\omega(e-\upsilon)]}$$

$$\le -\eta_1 |e|^2 - \eta_2 \left|\frac{e}{1+\exp[-\omega(e-\upsilon)]}\right| + \eta_1 |e||\varepsilon(\boldsymbol{w})| + \eta_2 \left|\frac{1}{1+\exp[-\omega(e-\upsilon)]}\right||\varepsilon(\boldsymbol{w})|$$

$$= (-\eta_1 |e| - \eta_2 \left|\frac{1}{1+\exp[-\omega(e-\upsilon)]}\right|)(|e| - |\varepsilon(\boldsymbol{w})|) \tag{3-20}$$

If $e < 0$,

$$\dot{V} = (-\eta_1 e - \eta_2 (-\frac{1}{1+\exp[\omega(e+\upsilon)]}))(e - \varepsilon(\boldsymbol{w}))$$

$$= -\eta_1 |e|^2 - \eta_2 \left|-\frac{e}{1+\exp[\omega(e+\upsilon)]}\right| + \eta_1 e\varepsilon(\boldsymbol{w}) + \eta_2 (-\frac{\varepsilon(\boldsymbol{w})}{1+\exp[\omega(e+\upsilon)]})$$

$$\le -\eta_1 |e|^2 - \eta_2 \left|-\frac{e}{1+\exp[\omega(e+\upsilon)]}\right| + \eta_1 |e||\varepsilon(\boldsymbol{w})| + \eta_2 \left|-\frac{1}{1+\exp[\omega(e+\upsilon)]}\right||\varepsilon(\boldsymbol{w})|$$

$$= (-\eta_1 |e| - \eta_2 \left| - \frac{1}{1+\exp[\omega(e+\upsilon)]} \right|)(|e| - |\varepsilon(\boldsymbol{w})|) \tag{3-21}$$

From Equations (3-20) and (3-21), if $|e| > |\varepsilon(\boldsymbol{w})|$, then $\dot{V} < 0$ for both situations of $e > 0$ and $e < 0$; it means that the estimated weights of the NF approximator will converge to minima until $|e| \le |\varepsilon(\boldsymbol{w})| \le \varepsilon_M$. Because $\varepsilon(\boldsymbol{w})$ is an optimal approximation error, it is seen that the NF approximator will asymptotically converge until the desired approximation accuracy is obtained. When $e = 0$, $T(x) = 0$, and then the EG becomes the gradient descent algorithm. Therefore, it can be concluded that the proposed EG method is Lyapunov stable.

## 3.3 The Modified LSE

### 3.3.1 The Classical LSE

In the general least-squares problem, the output of a linear model *y* is given by the linearly parameterized expression

$$y = \theta_1 f_1(\boldsymbol{u}) + \theta_2 f_2(\boldsymbol{u}) + ...\theta_n f_n(\boldsymbol{u}) \tag{3-22}$$

where $\boldsymbol{u} = [u_1,...,u_p]^T$ is the model's input vector, $f_1,...,f_n$ are known functions of *u*, and $\theta_1,...,\theta_n$ are unknown parameters to be estimated.

In statistics, the task of fitting data using a linear model is referred to as linear regression. Thus Equation (3-22) is also called the regression function, and $\theta_i$ are called regression coefficients.

To identify the unknown parameters $\theta_i$, usually we have to perform experiments to obtain a training data set composed of data pairs $\{(u_i; y_i), i = 1,..., m\}$; they represent desired input-output pairs of the target system to be modeled. Substituting each data pair into Equation (3-22) yields *m* linear equations:

$$\begin{cases} f_1(u_1)\theta_1 + f_2(u_1)\theta_2 + ... + f_n(u_1)\theta_n = y_1 \\ f_1(u_2)\theta_1 + f_2(u_2)\theta_2 + ... + f_n(u_2)\theta_n = y_2 \\ \vdots \\ f_1(u_m)\theta_1 + f_2(u_m)\theta_2 + ... + f_n(u_m)\theta_n = y_m \end{cases} \tag{3-23}$$

Using matrix notation, we can rewrite the preceding equations in a concise form:

$$\boldsymbol{A\theta} = \boldsymbol{y} \tag{3-24}$$

where $\boldsymbol{A}$ is an $m \times n$ matrix.

Now, instead of finding the exact solution to Equation (3-24), we want to search for $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$ to minimize the sum of squared error defined by

$$\begin{aligned} E(\boldsymbol{\theta}) &= \sum_{i=1}^{m} \left(y_i - a_i^T \boldsymbol{\theta}\right)^2 \\ &= e^T e \\ &= \left(\boldsymbol{y} - \boldsymbol{A\theta}\right)^T \left(\boldsymbol{y} - \boldsymbol{A\theta}\right) \end{aligned} \tag{3-25}$$

where $\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{A\theta}$ is the error vector produced by a specific choice of $\boldsymbol{\theta}$. Note that $E(\boldsymbol{\theta})$ is in quadratic form and has a unique minimum at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$. The following theorem states a necessary condition satisfied by the LSE.

If $\boldsymbol{A}^T\boldsymbol{A}$ is non-singular, $\hat{\boldsymbol{\theta}}$ is unique and is given by

$$\hat{\boldsymbol{\theta}} = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y} \tag{3-26}$$

To facilitate the description of new technique, recursive least squares identification is briefly addressed next.

First, the LSE derived above can be expressed as

$$\boldsymbol{\theta_k} = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y} \tag{3-27}$$

Obviously, $\boldsymbol{\theta}_{k+1}$ can be expressed as

$$\boldsymbol{\theta}_{k+1} = \left(\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{a}_{k+1}^T\end{bmatrix}^T\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{a}_{k+1}^T\end{bmatrix}\right)^{-1}\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{a}_{k+1}^T\end{bmatrix}^T\begin{bmatrix}\boldsymbol{y}\\ y_{k+1}\end{bmatrix} \tag{3-28}$$

where $\boldsymbol{A}$ is a matrix contains old data and $\boldsymbol{a}_{k+1}$ is data vector at time step *k+1*. To simplify the notation, we introduce two $n \times n$ matrices $\boldsymbol{P}_k$ and $\boldsymbol{P}_{k+1}$ defined by

$$\boldsymbol{P}_k = (\boldsymbol{A}^T\boldsymbol{A})^{-1} \tag{3-29}$$

$$\boldsymbol{P}_{k+1} = \left( \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{a}_{k+1}^T \end{bmatrix}^{\boldsymbol{T}} \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{a}_{k+1}^T \end{bmatrix} \right)^{-1}$$

$$= \left( \begin{bmatrix} \boldsymbol{A}^{\boldsymbol{T}} & \boldsymbol{a}_{k+1} \end{bmatrix} \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{a}_{k+1}^T \end{bmatrix} \right)^{-1}$$

$$= (\boldsymbol{A}^T \boldsymbol{A} + \boldsymbol{a}_{k+1} \boldsymbol{a}_{k+1}^T)^{-1} \tag{3-30}$$

Then we have

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_{\boldsymbol{k}} - \frac{\boldsymbol{P}_{\boldsymbol{k}} \boldsymbol{a}_{\boldsymbol{k+1}} \boldsymbol{a}_{\boldsymbol{k+1}}^{\boldsymbol{T}} \boldsymbol{P}_{\boldsymbol{k}}}{\boldsymbol{1} + \boldsymbol{a}_{\boldsymbol{k+1}}^{\boldsymbol{T}} \boldsymbol{P}_{\boldsymbol{k}} \boldsymbol{a}_{\boldsymbol{k+1}}} \tag{3-31}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{P}_{k+1} \boldsymbol{a}_{k+1} (y_{k+1} - \boldsymbol{a}_{k+1}^T \boldsymbol{\theta}_k) \tag{3-32}$$

### 3.3.2 The Developed Modified LSE

The modified LSE method is suggested in this work to update the linear parameters of the NF approximator. We introduce two $n \times n$ matrices $P_{t-1}$ and $P_t$, where $n$ is the number of linear parameters. The corresponding update law is represented as

$$\widetilde{\boldsymbol{\psi}}_t = \boldsymbol{P}_t \boldsymbol{a}_t (e_y) \tag{3-33}$$

where $\widetilde{\boldsymbol{\psi}}_t$ is learning step in each training update process.

$$\boldsymbol{P}_t = \boldsymbol{P}_{t-1} - \frac{\boldsymbol{P}_{t-1} \boldsymbol{a}_t \boldsymbol{a}_t^T \boldsymbol{P}_{t-1}}{1 + \boldsymbol{a}_t^T \boldsymbol{P}_{t-1} \boldsymbol{a}_t} \tag{3-34}$$

$$e_y = -\widetilde{y}$$

$$= y_d - y \tag{3-35}$$

$$y = \boldsymbol{a}_t^T \boldsymbol{\psi}_{t-1} \tag{3-36}$$

In the above expressions, $\widetilde{\boldsymbol{\psi}}_t = \boldsymbol{\psi}_t - \boldsymbol{\psi}_{t-1}$ is the increment of target update vector and $\boldsymbol{\psi}_t$ is a vector of linear parameters. $\boldsymbol{a}_t$ is the firing strength vector which is the multiplier of its corresponding linear parameter vector $\boldsymbol{\psi}_t$.

It is known that the update of $\boldsymbol{\psi}_t$ in Equation (3-33) will become very slow as $e_y$ approaches zero. The modified LSE method is proposed to solve this problem in the classical LSE. To increase the learning step $\widetilde{\boldsymbol{\psi}}_t$, the modified LSE will use a function $T(.)$ to speed up convergence:

$$\widetilde{\boldsymbol{\psi}}_t = \boldsymbol{P}_t \boldsymbol{a}_t (e_y + T(e_y)) \tag{3-37}$$

Comparing Equations (3-37) and (3-33), the increment of target update vector is enlarged when $e_y$ decreases into the work region of $T(.)$. Because $T(.)$ starts to contribute to fast convergence at this time. The work region of $T(.)$ is usually defined near zero. By introducing $T(.)$ in the modified LSE, the convergence speed can be improved compared with the classic LSE, as illustrated in Figure 3.3, especially as error is small.

### 3.3.3 Stability Analysis of the Modified LSE Method

The Lyapunov stability of the proposed modified LSE is proven as follows. Assume that $\boldsymbol{P}_0 = \alpha \boldsymbol{I}$, where $\boldsymbol{I}$ is an identity matrix; $\alpha$ is a sufficiently large positive number to guarantee that $\boldsymbol{P}_t$ is positive semi-definite over a sufficiently long period. According to the definition of positive semi-definite, it is seen that $\boldsymbol{a}_t^T \boldsymbol{P}_t \boldsymbol{a}_t \in R$ and $\boldsymbol{a}_t^T \boldsymbol{P}_t \boldsymbol{a}_t \geq 0$. From Equations (3-34)-(3-37),

$$\dot{y} = \boldsymbol{a}_t^T \dot{\boldsymbol{\psi}}_{t-1}$$

$$= \boldsymbol{a}_t^T \frac{\boldsymbol{\psi}_t - \boldsymbol{\psi}_{t-1}}{\Delta t}$$

$$= \boldsymbol{a}_t^T \boldsymbol{P}_t \boldsymbol{a}_t (e_y + T(e_y))\Delta t^{-1} \tag{3-38}$$

$$\dot{\tilde{y}} = \dot{y} - \dot{y}_d$$

$$= \dot{y}$$

$$= a_t^T P_t a_t (e_y + T(e_y))\Delta t^{-1} \tag{3-39}$$

If the Lyapunov function is selected as

$$V = \frac{1}{2}\tilde{y}^T \tilde{y} \tag{3-40}$$

Its time derivative becomes:

$$\dot{V} = \tilde{y}^T \dot{\tilde{y}}$$

$$= \tilde{y}^T \boldsymbol{a}_t^T \boldsymbol{P}_t \boldsymbol{a}_t (e_y + T(e_y))\Delta t^{-1}$$

$$= -e^T \boldsymbol{a}_t^T \boldsymbol{P}_t \boldsymbol{a}_t (e_y + T(e_y))\Delta t^{-1}$$

$$= \left|\boldsymbol{a}_t^T \boldsymbol{P}_t \boldsymbol{a}_t\right|(-\left|e_y\right|^2 - \left|e_y T(e_y)\right|)\Delta t^{-1} \le 0 \tag{3-41}$$

Then it is concluded that the proposed modified LSE method is Lyapunov stable.

## 3.4 Hybrid Training Process

Although we can apply backpropagation to identify the parameters in an adaptive network, this simple optimization method usually takes a long time before it converges. Since the NF approximator consists of both linear and nonlinear system parameters, a hybrid training

techniques based on the proposed EG method and modified LSE will be used for system training. Each training update process takes two runs: in the backward pass, the nonlinear parameters of the NF approximator are updated by the developed EG method. In the forward pass, the linear parameters of the NF approximator are trained using the modified LSE technique. The modified LSE can reduce the search space dimension and thus the overall training speed can be improved. Furthermore, since each training update process consists of two independent learning processes which take different initial condition, it is possible to escape more local minima in training. Test results of the proposed techniques will be discussed in the Chapter 5.

# Chapter 4 The Enhanced Sliding Mode Control

## 4.1 Experimental Setup

To facilitate the description of the developed new control techniques, the experimental setup used in this research will be described first. As shown in Figure 4.1, the experimental setup consists of a flexible beam, actuator, power amplifier, PCI terminal board, sensors, motor and computer. The flexible beam is clamped vertically at one end; its another end is free and is connected with a servo motor that drives the actuator through a gearbox. The actuator includes a rigid beam and a cross beam. Magnetic blocks are attached to different positions on the flexible

beam to test the robustness of the controller. The deflection of the flexible beam is measured by a strain gauge connected at the bottom of the flexible beam. An encoder is connected to the rigid beam to measure its angular position. The PCI terminal board performs AD/DA operations for system measurement and control [59]. Both the control signal and disturbance are generated from a computer. The properties of these related components in this experimental setup are summarized as follows:

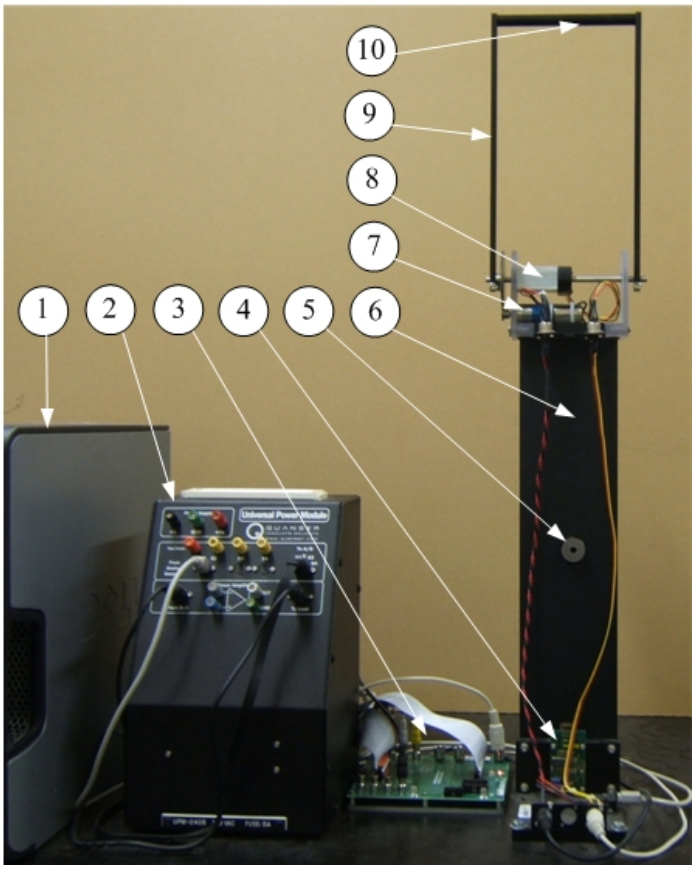


Figure 4.1. The experimental setup: (1)-computer, (2)-power amplifier, (3)-PCI terminal board, (4)-strain gauge, (5)-blocks, (6)-flexible beam, (7)- motor, (8)-encoder, (9)-rigid beam (actuator), (10)-cross beam.

- Flexible beam: The beam is instrumented with a strain gauge which is calibrated to give

1 *volt* per 2.54 *cm*, to indirectly measure vibration. The tested beam is a steel sheet which can be in different dimension, material, and orientation.

- Motor (Series 2338S006): The motor drives the load (rigid beam and cross beam) through a gear ratio of 70:1.
- The encoder (SRV02): The encoder measures the rotation angle by using a 1024 count disc which in quadrature results in 4096 *counts/rev*.
- Strain gauge (A9-232-0): The strain gauge measures the beam deflection and sends the signal to the universal power module.
- Inertia load is applied by two rigid beams and a round cross beam.
- Universal power module (UPM-2405): The power module is a power amplifier to drive the actuator. The power module consists of one power supply (±12 Volt), four analog sensor inputs, and one power amplified analog output.
- DSP board (A11-368-3): It is a Quanser Q4 terminal board which contains the general A/D and D/A converters. All of the connectors from the universal power module are connected to the DSP board by wires.

It should be stated that the experimental setup as described in Figure 4.1 is a universal workstation which can be used for R&D in vibration control in different systems such as robot arms, towers of wind turbines, tall buildings, etc.

## 4.2 Mathematical Model

The flexible beam structure as illustrated in Figure 4.1 can be modeled as shown in Figure 4.2. The system model can be described by the following dynamical equations:

$$(M + m_p)\ddot{x} - m_p c \sin(\theta)\dot{\theta}^2 + m_p c \cos(\theta)\ddot{\theta} + Kx = 0 \tag{4-1}$$

$$m_p c \cos(\theta)\ddot{x} + (m_p c^2 + I)\ddot{\theta} = T \tag{4-2}$$

$$T = \frac{K_g K_m}{R_m}(V - K_g K_m \dot{\theta}) \tag{4-3}$$

where $x$ is the deflection of the flexible beam in metres; $\theta$ in rad is the rotation of the rigid beam with respect to the vertical position, which varies with time $t$; $T$ and $V$ are the respective torque and control signals; $M$ is the mass of the motor and its fixture; $m_p$ is the mass of the cross beam mass; $c$ is the length of the rigid beam; $K$ represents the effective stiffness of the flexible beam along horizontal direction; $I$ and $R_m$ are the rigid beam inertia and motor armature resistance, respectively; $K_g$ and $K_m$ are the respective gear ratio and motor torque constant. These parameters of the model are listed in Table 4.1.

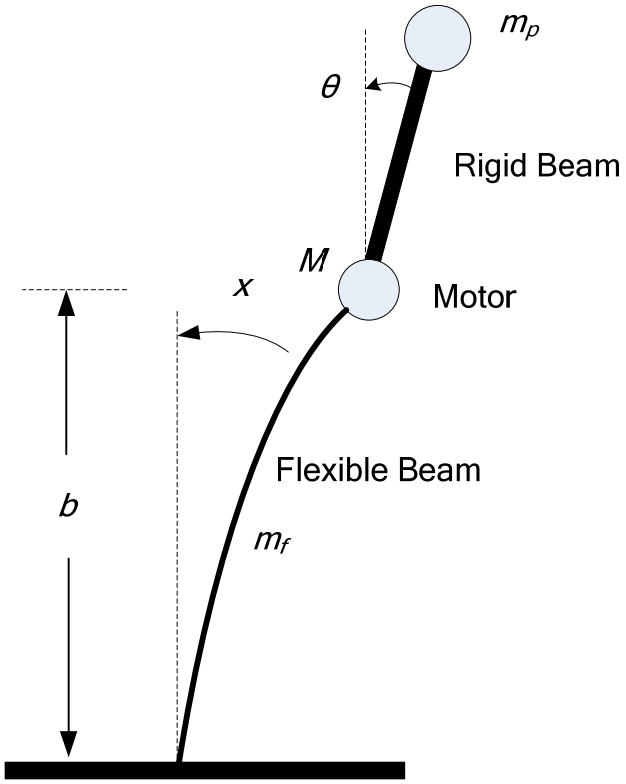


Figure 4.2. A dynamic model of flexible structure system.

Table 4.1: Parametric values for the flexible structure.

| Physical parameters | Symbol | Value/Units |
|---|---|---|
| Mass of the motor and its fixture | $M$ | 0.6 Kg |
| Cross beam mass | $m_p$ | 0.05 Kg |
| Rigid beam inertia | $I$ | 0.0039 Kg $m^2$ |
| Rigid beam length | $c$ | 0.285 m |
| Effective stiffness of the flexible beam along x direction | $K$ | 30 N/m |
| Motor Torque constant | $Km$ | 0.0767 Nm/Amp |
| Motor Armature resistance | $Rm$ | 2.6 Ohm |
| Flexible beam length | $b$ | 0.44 m |
| Flexible beam mass | $m_f$ | 0.22 Kg |
| Total gear ratio | $Kg$ | 70 |
| Nominal voltage | $U_N$ | 6 Volt |
| No-load current | $I_o$ | 0.08A |
| Stall torque | $M_H$ | 2.42 oz-in |
| Friction torque | $M_R$ | 0.086 oz-in |

To simplify the following presentation, $\{x_1, x_2, x_3, x_4\}$ are used to denote system states $\{\theta, x, \dot{\theta}, \dot{x}\}$ which represent the angle of the rigid beam, the deflection of the flexible beam, the angular velocity of the rigid beam and the deflection velocity of the flexible beam, respectively. The control objective is to eliminate the vibration in the flexible structure.

## 4.3 The Enhanced Sliding Mode Control

As discussed in Chapter 1, the fast TSM control can speed up the rate of convergence near the equilibrium point; however it has implementation problems because the initial condition

cannot be zero. To overcome these implementation problems while sustain its advantages in fast convergence, an enhanced sliding mode (ESM) control is developed in this chapter, discussed as follows:

The sliding surface of the proposed ESM controller is defined as

$$S_1 = x_4 + c_2(x_2) + c_3 T(x_2) \tag{4-4}$$

where $T(.)$ is a new function to speed up convergence, which is defined as

$$T(x) = \begin{cases} \dfrac{1}{1+\exp[-\omega(x-\upsilon)]} & if\ x > 0 \\ 0 & if\ x = 0 \\ -\dfrac{1}{1+\exp[\omega(x+\upsilon)]} & if\ x < 0 \end{cases} \tag{4-5}$$

where $\omega > 0$ and $\upsilon > 0$ are constants.

As a comparison, the sliding surface of the classical SM control is represented as

$$S_2 = x_4 + c_2(x_2) \tag{4-6}$$

Consider a simple nonlinear system:

$$\dot{x} = f(x) + u, \tag{4-7}$$

where $|f(x)| < L$ is an unknown but bounded continuous function; $u \in R$ is the control input of the system.

The sliding mode of the ESM control is given as:

$$\dot{x} = -\alpha_1 x - \beta_1 T(x) \tag{4-8}$$

where $\alpha_1$ and $\beta_1$ are constants. The corresponding control law is designed as

$$u = -\hat{f}(x) - \alpha_1 x - \beta_1 T(x) \tag{4-9}$$

where $\hat{f}(x)$ is the NF approximator which is used to compensate for the unknown nonlinear system functions. This time the *sign(.)* is removed. The other items in Equation (4-9) aim at achieving a sliding mode for system stabilization and faster convergence which will be discussed later. Then the closed-loop system becomes

$$\begin{aligned} \dot{x} &= f(x) - \hat{f}(x) - \alpha_1 x - \beta_1 T(x) \\ &= \tilde{f}(x) - \alpha_1 x - \beta_1 T(x) \end{aligned} \tag{4-10}$$

where $\tilde{f}(x) = f(x) - \hat{f}(x)$ is the NF approximator error. As discussed in Chapter 3, the NF system can converge to an optimal approximation with a small bounded residual error, that is to say, the unknown function $f(x)$ can be approximated with a small bounded error tolerance. Figure 3.3 shows an example of $y = T(x)$ and $y = x$ when $\omega$=15 and $\upsilon$= 0.5.

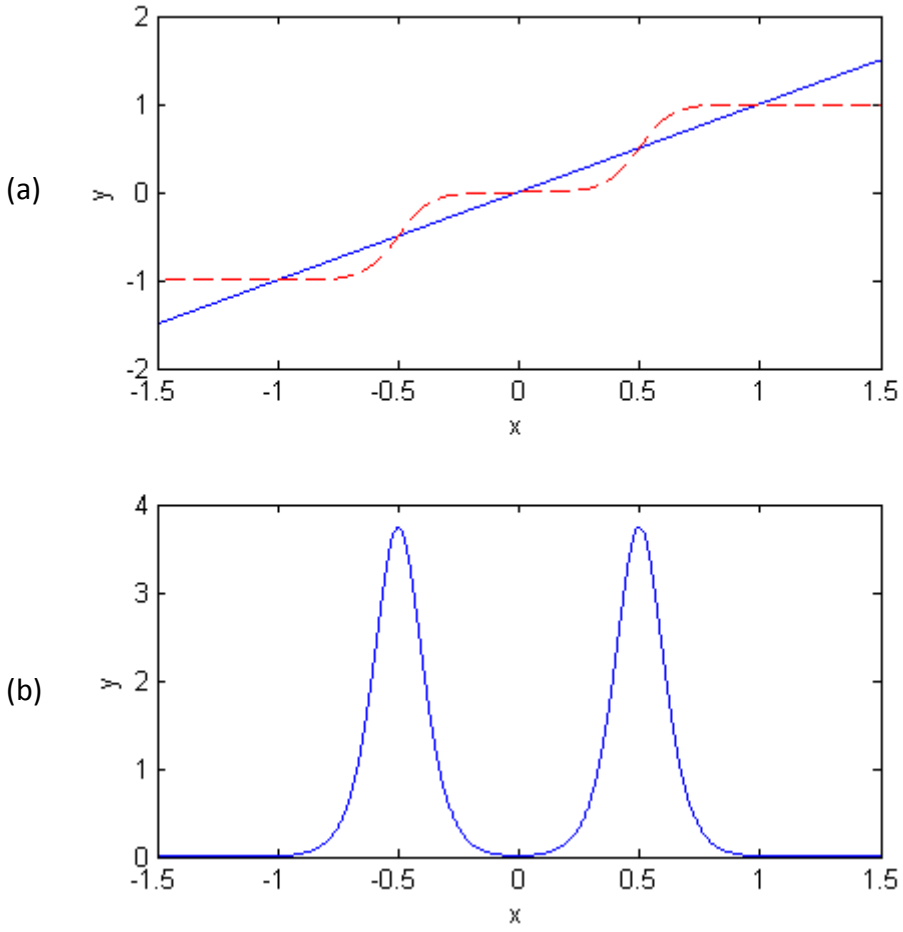


Figure 4.3 (a) Graph of $y=T(x)$ (dashed line) and $y = x$ (solid line). (b) Graph of function $y=dT(x) / dx$.

From some systematic investigation, it is found that, for terminal attractor, when system state is very close to the origin, strong control action variation is generated, which may, in turn, cause implementation failure. Such a problem will be solved using the following close-loop system representation.

When the state is very small (e.g., $x \in (-0.2,0.2)$ in this case) or very large (e.g., $x \in (-\infty,-0.8) \cup (0.8,\infty)$ in this case), the close-loop system can be described by

$$\dot{x} = -\alpha_1 x \tag{4-11}$$

However the state convergence will be slow when the state is over the middle range (e.g., $x \in (-0.8,-0.2) \cup (0.2,0.8)$ ). By applying *T(.)* so as to improve convergence over the middle range, the system can be modeled as

$$\dot{x} = -\beta_1 T(x)\,. \tag{4-12}$$

The accelerating effects of $T(.)$ can be demonstrated by using its derivative functions:

$$\frac{dT(x)}{dx} = \begin{cases} \dfrac{\omega \exp(-\omega(x-\upsilon))}{[1+\exp(-\omega(x-\upsilon))]^2} & if \quad x > 0 \\ 0 & if \quad x = 0 \\ \dfrac{\omega \exp(\omega(x+\upsilon))}{[1+\exp(\omega(x+\upsilon))]^2} & if \quad x < 0 \end{cases} \tag{4-13}$$

Since the denominator of Equation (4-13) and the exponential function in the numerator are non-negative given $\omega > 0$ as stated before, we have concluded $\frac{dT(x)}{dx} \geq 0$ for all $x \in R$, as demonstrated in Figure 4.3(b). When $x$ is far away from the origin or very close to the origin, there is no apparent promotion on speed improvement. In the middle range of the error such as $(-0.8,-0.2) \cup (0.2,0.8)$ ), however, the promotion effect becomes significant. The optimal promotion can be achieved by tuning the parameters $\omega$ and $\upsilon$.

It should be stated that the developed ESM control is inherently a computer-based discrete-time control. Obviously, the apparent difference between discrete-time system and continuous-time system is related to the limited switching speed of the switching control item *sign(.)*. In the ESM system, the switching control item is replaced by an NF approximator as discussed in

Chapter 3. Thus the control errors associated with the discrete-time control signal (which can also be regarded as part of control error) can be reduced by the compensation effect of the adaptive NF system and by fine-tuning the initial conditions of the NF system parameters. Furthermore, as demonstrated in Chapter 1, the chattering problem in the classical SM control can also be prevented by using the NF approximator instead of switching control item and to make the state maintain on the defined sliding surface.

# Chapter 5 Performance Evaluation of ESM Control and Training Techniques

## 5.1 Overview

To verify the effectiveness of the developed ESM controller and the related training techniques (i.e., EG and modified LSE), a series of tests will be conducted with the experimental setup as shown in Figure 4.1. The proposed ESM controller with suggested hybrid training technique with EG and modified LSE, specified by Controller-1, is first properly implemented on the test workstation. To make a comparison, test results from the related controllers are also listed:

- Controller-2: The ESM controller with a hybrid training technique but based on the classic gradient algorithm and modified LSE method. Its purpose is to test the effectiveness of the proposed EG method.

- Controller-3: The ESM controller with a hybrid training technique but based on proposed EG and the classic LSE method. Its purpose is to test the effectiveness of the proposed modified LSE technique.

- Controller-4: An SM controller (including the same NF approximator as in ESM) with the suggested hybrid training technique (EG and modified LSE). Its goal is to test the effectiveness of the developed ESM controller.

In each test, the flexible beam is excited by a disturbance generated by the DC motor to drive the rigid beam to a target position (± 20 deg versus the vertical position). Since the natural

frequency of the flexible beam is about 1.136 Hz or with the period of 0.88 sec, as is shown in Figure 5.1.

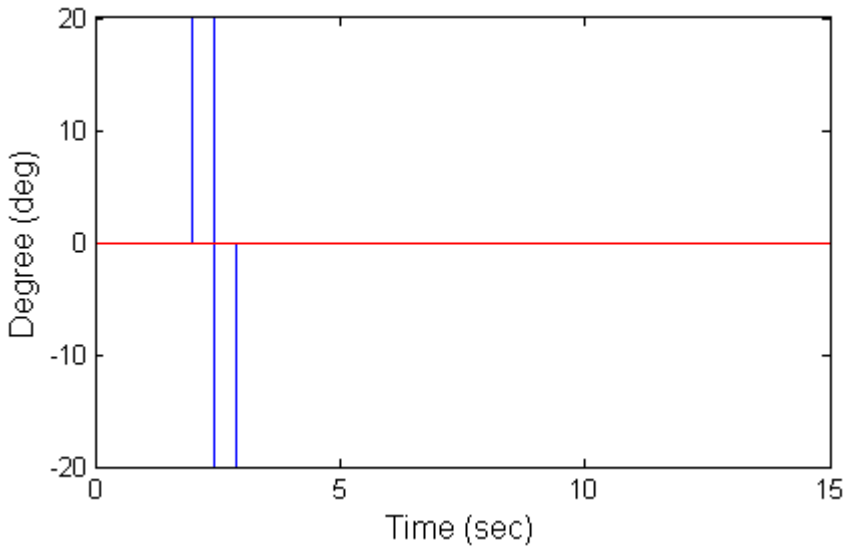


Figure 5.1. The disturbance

Firstly we run the experiments with no extra mass blocks attached. Figure 5.2 shows the control performance of the related four controllers over 15 seconds. The performance comparison is based on the measurements of overshoot and settling time with respect to the mean steady state. Since the tested beam is very flexible (with a first-mode natural frequency 1.136 Hz only), some high frequency vibration remains in the steady state. The settling time is measured as soon as the deflection amplitude becomes lower than 0.2 cm. Table 5.1 summarizes the test results from different controllers.

(a)

(b)

(c)

(d)

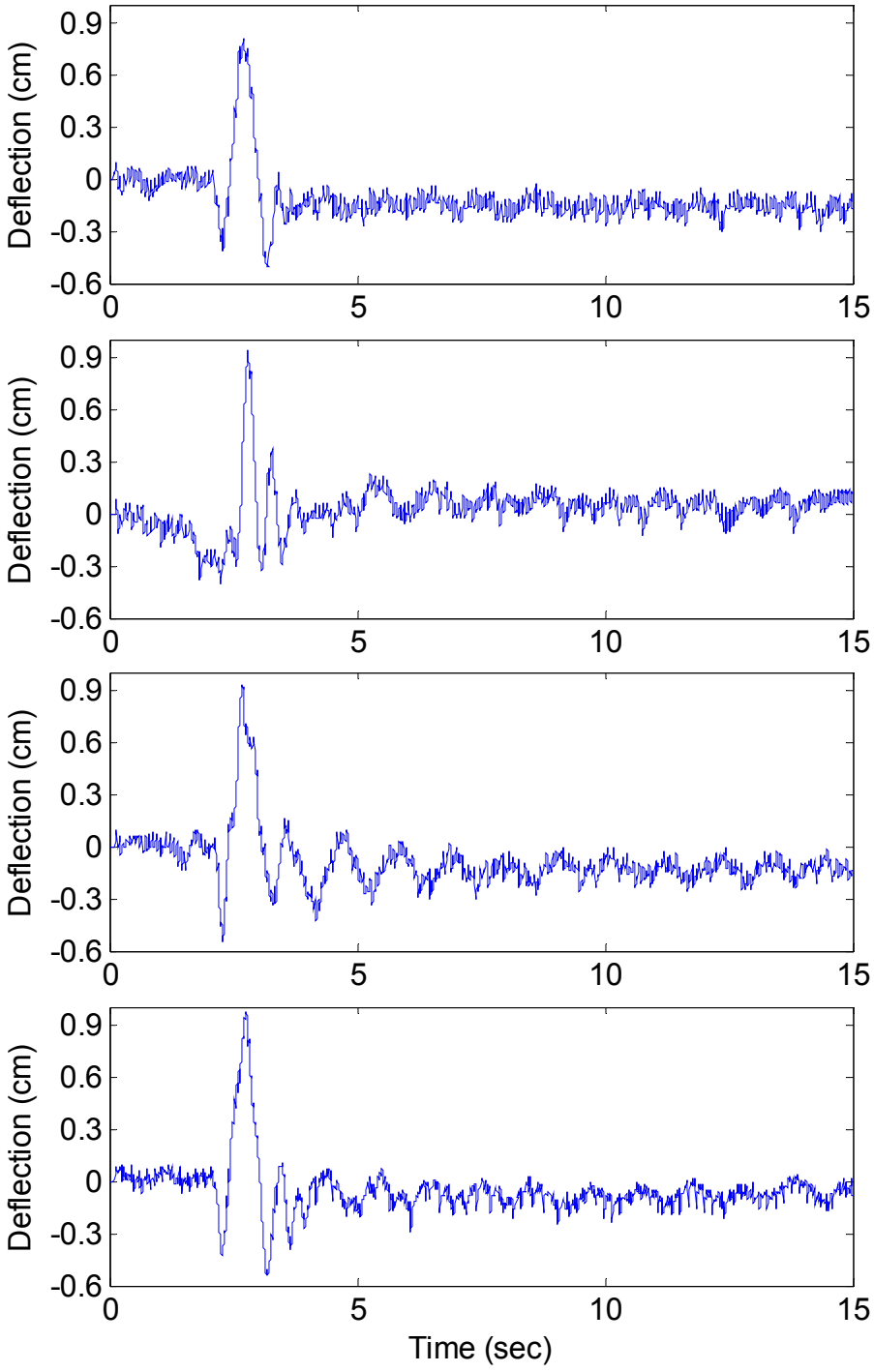


Figure 5.2. The deflection of the flexible beam by using different controllers: (a) the proposed Controller-1, (b) Controller-2, (c) Controller-3, (d) Controller-4

Table 5.1: The experiment results of the related controllers without extra mass blocks attached.

| Controller | Overshoot | Settling Time |
|---|---|---|
| Controller-1 | 0.76 cm | 3.4 s |
| Controller-2 | 0.93 cm | 6.8 s |
| Controller-3 | 0.92 cm | 7.4 s |
| Controller-4 | 0.97 cm | 6.1 s |

The only difference between Controller-1 and Controller-2 is related to the suggested EG method versus gradient algorithm. The overshoot of Controller-1 is about 82% of that of Controller-2. Moreover the settling time of Controller-1 takes only half of that of Controller-2. It is seen that the proposed EG method can improve system's performance and convergence effectively compared with the gradient algorithm.

The comparison between Controller-1 and Controller-3 is to evaluate the proposed modified LSE technique versus the classical LSE. It is seen that Controller-1 outperforms Controller-3 with respect of both criteria (i.e. overshoot and settling time). For example, Controller-1 takes only 54% of the settling time used in Controller-3. Apparently, the suggested modified LSE technique is superior to the classical LSE in training convergence.

Both Controller-1 and Controller-4 are tuned using the same hybrid training technique; the comparison between them is to examine the performance of the ESM controller versus a classical SM control. Similarly, Controller-1 provides better performance than Controller-4 with respect to overshoot and settling time. For example, Controller-1 can reduce 22% of the overshoot of

Controller-4. On the other hand, it should be stated that the performance gap between the developed ESM control (Controller-1) and SM control (Controller-4) should be even larger. Because the SM system is also trained by the proposed hybrid training technique, the relatively poor performance of the SM can be compensated to some extent by the effective training operations.

## 5.2 Robustness Test

To verify the robustness of the developed ESM control strategy, two adhesive mass blocks are attached to the flexible beam in three different locations during the tests: a top position about 10 cm below the top end of the flexible beam, a middle position (as shown in Figure 4.1), and a bottom position about 10 cm above the bottom end of the flexible beam. Each block weighs about 100 g. When the blocks are attached to the flexible beam, its system dynamics varies (regarding to the overall mass and the natural frequencies). Correspondingly the variable $K$ in Equation (4-1) is changed. Different blocks positions correspond to different system dynamics.

When the blocks are placed at three different positions of the flexible beam, Figures 5.3-5.5 show the corresponding beam deflections from different controls, excited by the same square disturbance as shown in Figure 5.1. Tables 5.2-5.4 summarize the corresponding control performance.

(a)

(b)

(c)

(d)

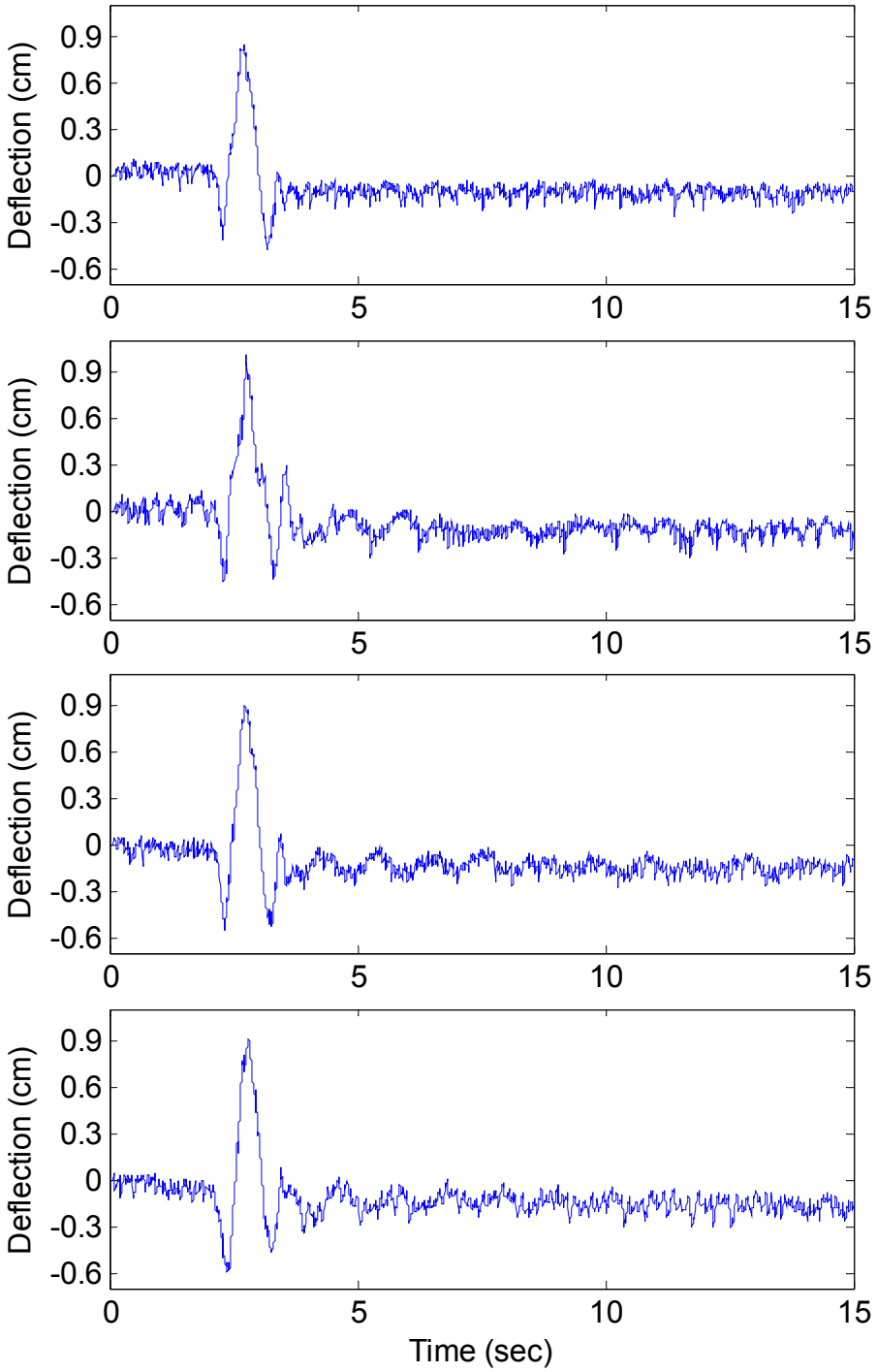


Figure 5.3. The deflection comparison of the flexible beam with extra mass blocks placed at a top position: (a) by Controller-1, (b) by Controller-2, (c) by Controller-3, and (d) by Controller-4.

(a)

(b)

(c)

(d)

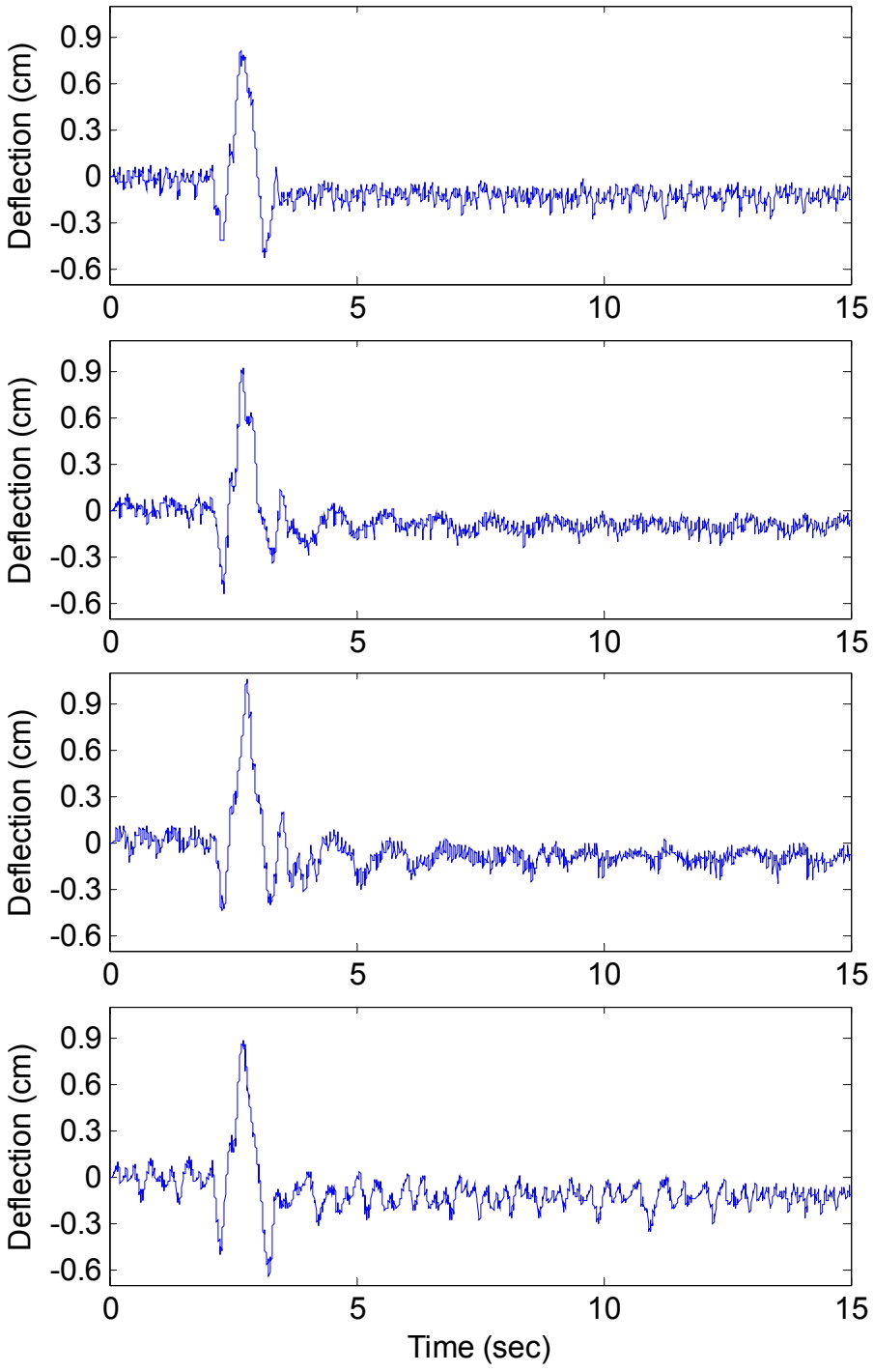


Figure 5.4. The deflection comparison of the flexible beam with extra mass blocks placed at a middle position: (a) by Controller-1, (b) by Controller-2, (c) by Controller-3, and (d) by Controller-4.

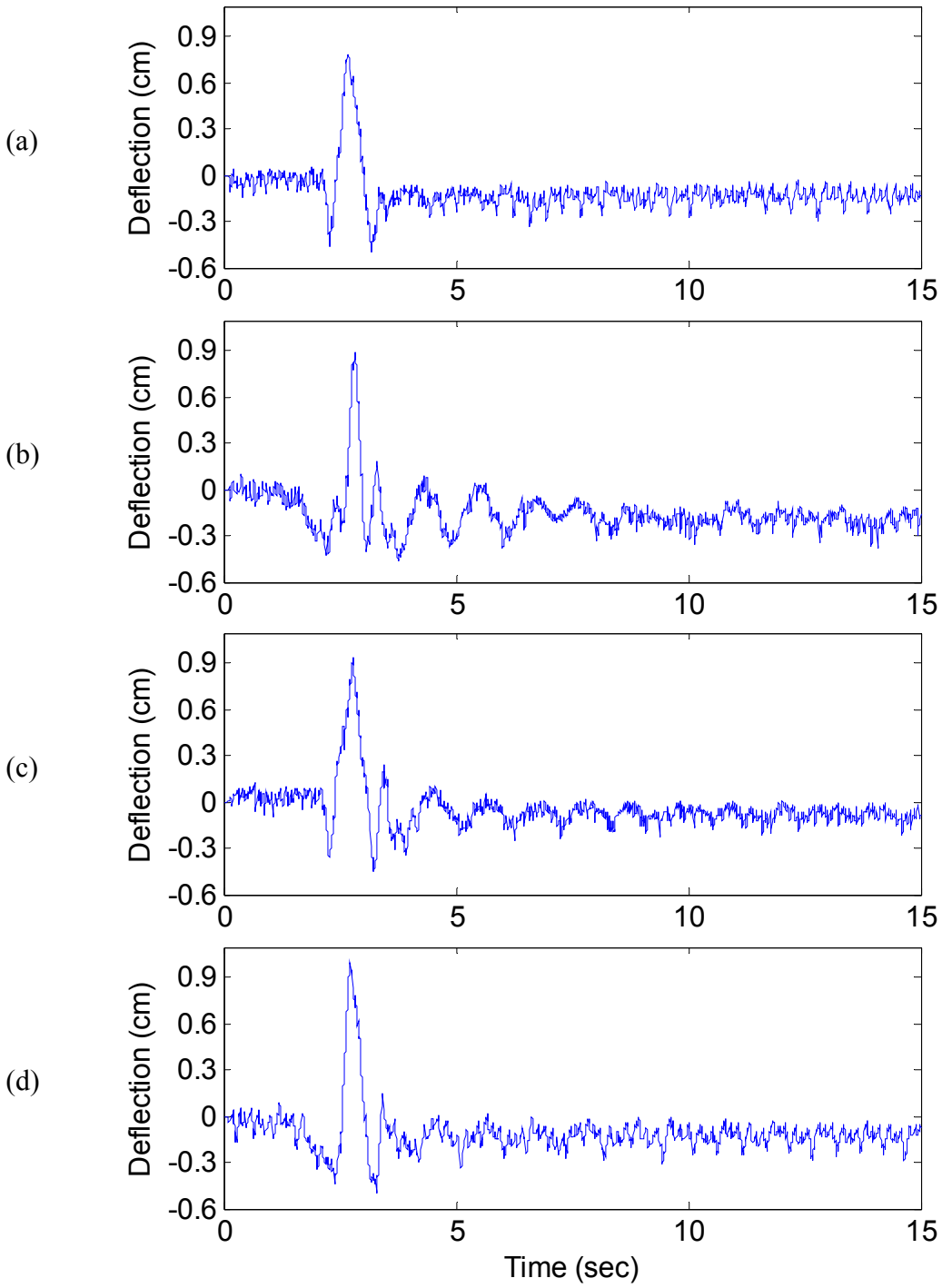


Figure 5.5. The deflection comparison of the flexible beam with extra mass blocks placed at a bottom position: (a) by Controller-1, (b) by Controller-2, (c) by Controller-3, and (d) by Controller-4.

Table 5.2: The experiment results of the related controllers with extra mass blocks placed at a top position

| Controller | Overshoot | Settling Time |
|---|---|---|
| Controller-1 | 0.82 cm | 3.5 s |
| Controller-2 | 1.00 cm | 6.7 s |
| Controller-3 | 0.90 cm | 8.2 s |
| Controller-4 | 0.91 cm | 6.0 s |

Table 5.3: The experiment results of the related controllers with extra mass blocks placed at a middle position

| Controller | Overshoot | Settling Time |
|---|---|---|
| Controller-1 | 0.80 cm | 3.4 s |
| Controller-2 | 0.92 cm | 7.1 s |
| Controller-3 | 1.00 cm | 6.6 s |
| Controller-4 | 0.89 cm | 12.3 s |

Table 5.4: The experiment results of the related controllers with extra mass blocks placed at a bottom position.

| Controller | Overshoot | Settling Time |
|---|---|---|
| Controller-1 | 0.78 cm | 3.4 s |
| Controller-2 | 0.89 cm | 8.0 s |
| Controller-3 | 0.93 cm | 9.1 s |
| Controller-4 | 0.99 cm | 8.2 s |

It can be seen that Controller-1 outperforms other related controllers. It is a robust control strategy. Among these three test scenarios, Controller-1 takes at most 50% of the settling time compared with other controllers due to its effective training convergence. It generates the least overshoot among these controllers in all three test scenarios because of the efficient ESM control strategy.

As a summary, a learning process can be regarded as the system model identification. The proposed training techniques enable controllers to effectively recognize the new system dynamics (parameters in this case) and accommodate different system conditions by universal approximation operations. The developed hybrid training technique can effectively improve training convergence and reduce the possible trapping due to local minima during the training process.

# Chapter 6 The Enhanced GA Technique

Another objective of this work is to develop a new search paradigm, called enhanced genetic algorithm (or EGA), to provide a more efficient global search strategy for optimization and system training. The EGA technique aims to promote the convergence and global search capability. It consists of a novel branch crossover operator and an enhanced MPT mutation operator. To facilitate illustration, some fundamental description of the classical GA is provided first.

## 6.1 Introduction to the Classical GA

GA is a derivative-free stochastic optimization method based loosely on the concepts of natural selection and evolutionary processes. GA has become a general purpose global optimization tool and has found many applications in different fields because of its specific characteristics:

1. GA is a parallel search algorithm that can be implemented on parallel processing operations.

2. It is applicable to both continuous and discrete optimization problems.

3. It is stochastic and less likely to get trapped in local minima than other classical training methods such as gradient algorithm and LSE.

4. Its flexibility facilitates both structure and parameter identification in complex models such as NNs and fuzzy inference systems.

GA encodes each point in a parameter space into a binary bit string called a chromosome, and each point is associated with a fitness value (which is usually equal to the objective function evaluated at the point). Instead of a single point, GA usually keeps a set of points as a population, which is then evolved repeatedly toward a better overall fitness value [37]. In each generation, the GA constructs a new population using genetic operators such as crossover and mutation; members with higher fitness values are more likely to survive and to participate in mating operations. After a number of generations, the GA population will contain members with better fitness values. Consequently, GA is referred to as the population-based optimization that can improve performance by upgrading entire populations rather than individual members.

As illustrated in Figure 6.1, the evolutionary process in GA takes selection, crossover and mutation operations occurring in a single act of offspring generation, as described as follows.

Step 1: Initialize a population with randomly generated individuals and evaluate the fitness value of each individual.

Step 2: Select qualified members from the population to do crossover and mutation operations.

Step 3: Do crossover with a probability equal to the crossover rate.

Step 4: Perform mutation with a probability equal to the mutation rate.

Step 5: Repeat Step 2 to Step 4 until the stop criterion is satisfied.

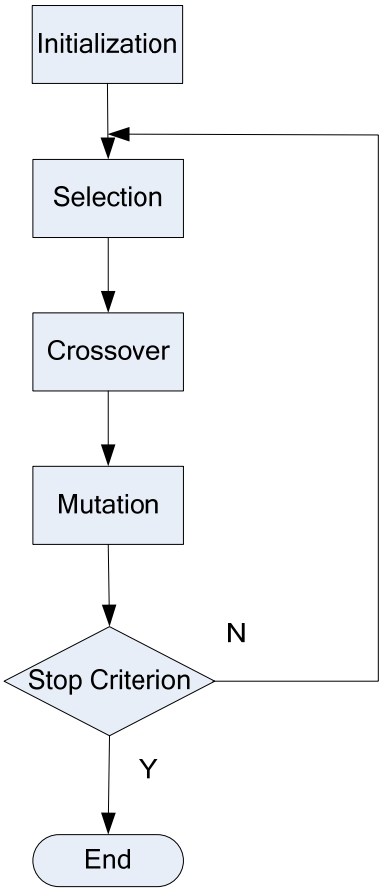


Figure 6.1 Flowchart of the classical GA.

The main processes are discussed as follows:

**1) Selection:** The selection operation determines which parents participate in producing offspring for the next generation. Usually members are selected for mating with a selection probability proportional to their fitness values. The effect of this selection method is to allow members with higher fitness values to reproduce and replace members with lower fitness values.

**2) Crossover:** To exploit the potential of the current gene pool, crossover operators are used to generate new chromosomes that can retain good features from the previous generation. Crossover is usually applied to select pairs of parents with a probability equal to a given

crossover rate. The effect of crossover is similar to that of mating in the natural evolutionary process, in which parents pass segments of their own chromosomes to their children. Therefore, some children are able to outperform their parents if they get good genes or genetic traits from both parents.

**3) Mutation:** Crossover exploits current gene potentials, but if the population does not contain all of the encoded information needed to solve a particular problem, no gene mixture can produce a satisfactory solution. For this reason, a mutation operator is used to generate new chromosomes. The common solution to implement mutation is to change parts of parents with a probability equal to a low mutation rate. A mutation operator can prevent all members from converging to a value throughout the entire population so as to prevent the population from stagnating at some local minima. The mutation rate is usually low so as to keep good chromosomes always in the population.

## 6.2 The Branch Crossover Operator

The Heuristic crossover (HC) operator is one of the commonly used classical crossover operators in GA methods. It generates a new member based on a linear combination of two parents, as illustrated in Figure 6.2. Its search direction is determined by the fitness of the parents and its search step is selected according to the distance between the parents. For example, given a pair of parents $x^{(1)} = (x_1^{(1)}, x_2^{(1)}, ..., x_n^{(1)})$ and $x^{(2)} = (x_1^{(2)}, x_2^{(2)}, ..., x_n^{(2)})$ with the fitness $\bar{x}^{(1)}$ and $\bar{x}^{(2)}$ where $\bar{x}^{(1)} > \bar{x}^{(2)}$, an offspring $y = (y_1, y_2, ..., y_n)$ is generated by

$$y_i = u(x_i^{(1)} - x_i^{(2)}) + x_i^{(1)} \tag{6-1}$$

where $u$ is a random number which is uniformly distributed over [0, 1].

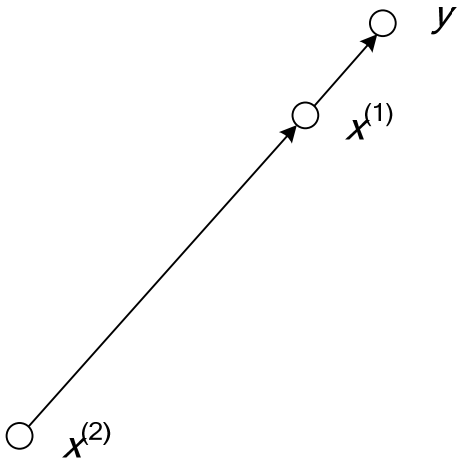


Figure 6.2. The crossover operation of the HC operator.

If the derived offspring lies beyond the feasible region, a new random number $u$ is generated to produce another offspring using Equation (6-1). If it fails to produce a feasible offspring after certain attempts, the HC operator randomly selects a point over the feasible region in place of the infeasible offspring produced. Accordingly, an offspring generated by HC will reduce the search dimension and lead to insufficient exploration of local minima.

The Laplace crossover (LC) operator is another classical crossover approach whose offspring are placed symmetrically over the positions of the parents. However, it explores local space in a stochastic fashion without the fitness guidance of parents, which will cause redundant calculations in local search operations and thus cost more execution time.

To speed up the search process and expand the local search space, a branch crossover (BC) operator is proposed in this work for local search. As demonstrated in Figure 6.3, three members are selected as a group: $x^{(1)} = (x_1^{(1)}, x_2^{(1)}, ..., x_n^{(1)})$ , $x^{(2)} = (x_1^{(2)}, x_2^{(2)}, ..., x_n^{(2)})$ and $x^{(3)} = (x_1^{(3)}, x_2^{(3)}, ..., x_n^{(3)})$, where $x^{(1)}$ and $x^{(3)}$ have the largest and smallest fitness among the three,

respectively; *n* is the dimension of the problem. The intermediate point $y^{(1)} = (y_1^{(1)}, y_2^{(1)}, ..., y_n^{(1)})$ is formulated by

$$y_i^{(1)} = u_1(x_i^{(1)} - x_i^{(2)}) + x_i^{(1)} \tag{6-2}$$

where $u_1$ is a random number uniformly distributed over [0, 1].

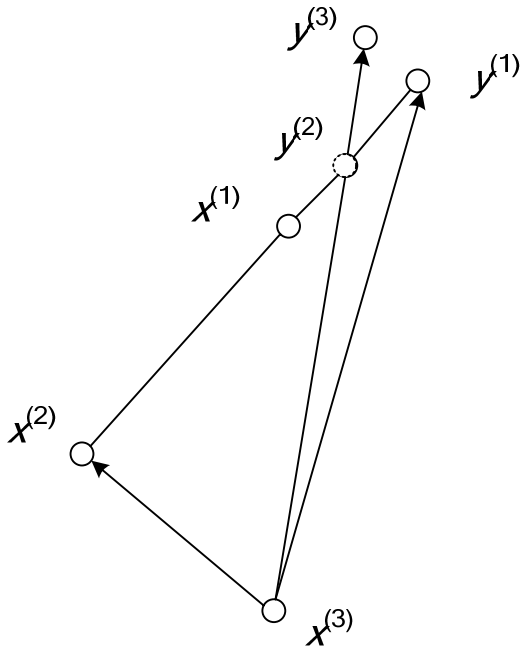


Figure 6.3. The crossover operation of the proposed BC operator.

The intermediate point $y^{(2)} = (y_1^{(2)}, y_2^{(2)}, ..., y_n^{(2)})$ is generated by

$$y_i^{(2)} = u_2(x_i^{(2)} - y_i^{(1)}) + y_i^{(1)} \tag{6-3}$$

where $u_2 = \dfrac{\bar{x}_2 - \bar{x}_3}{(\bar{x}_1 - \bar{x}_3) + (\bar{x}_2 - \bar{x}_3)}$; $\bar{x}_1$, $\bar{x}_2$ and $\bar{x}_3$ are the normalized fitness of $x^{(1)}$, $x^{(2)}$ and $x^{(3)}$, respectively. Since $x^{(1)}$ has a larger fitness than $x^{(2)}$ (i.e. $\bar{x}_1 > \bar{x}_2$) as stated before, it is seen that

$u_2 \in [0,0.5]$.

To facilitate implementation, let $u_2 = 0$, if $(\bar{x}_1 - \bar{x}_3) + (\bar{x}_2 - \bar{x}_3) = 0$. Thus the offspring $y^{(3)} = (y_1^{(3)}, y_2^{(3)}, ..., y_n^{(3)})$ will be formed as

$$y_i^{(3)} = u_3(y_i^{(2)} - x_i^{(3)}) + y_i^{(2)} \tag{6-4}$$

where $u_3 \in [0,1]$ is a constant.

The offspring generated by the proposed BC operator will be located around the best individual in the group (i.e. the three selected members); while the local search space can be further expanded. The method to derive $y$ in HC, as illustrated in Figure 6.2, may not be the optimal solution of the offspring. Our solution in the BC operator is to employ more variant factors to formulate an offspring. Based on the BC operator, the final offspring $y^{(3)}$ will be located on the extension of the segment from $x^{(3)}$ to $y^{(2)}$. With the help of the new variant factors, the local space will be explored more effectively. If the derived offspring lies beyond the feasible region, new random numbers $u_1$ and $u_3$ will be generated to produce another offspring using Equations (6-2)-(6-4). If it fails to produce a feasible offspring after some attempts, the BC operator will randomly select a point over the feasible region to substitute for the original infeasible offspring.

## 6.3 The EMM Mutation Operator

The commonly used classical mutation operators in constrained optimization include the non-uniform mutation (NUM) and Makinen-Periaux-Toivanen mutation (MPTM), but each having its own merits and limitations. For example, although MPTM explores the search space

more efficiently than NUM, NUM is more productive to evolve the search coverage than MPTM. For instance, given a point $x^t = (x_1^t, x_2^t, ..., x_n^t)$ , the NUM builds the mutated point $x^{t+1} = (x_1^{t+1}, x_2^{t+1}, ..., x_n^{t+1})$ by

$$x_i^{t+1} = \begin{cases} x_i^t + \eta(t, x_i^u - x_i^t), & \text{if } r \le 0.5 \\ x_i^t - \eta(t, x_i^t - x_i^l), & \text{otherwise} \end{cases} \tag{6-5}$$

where $t$ is the generation number and $r$ is a value randomly selected over [0,1]; $x_i^l$ and $x_i^u$ are the respective lower and upper bounds of the $i$th component of the decision vector. In addition,

$$\eta(t, y) = y\left(1 - \omega^{\left(1-\frac{t}{T}\right)^b}\right) \tag{6-6}$$

where $\omega$ is a random number uniformly distributed over [0,1]; $T$ is the maximum number of generations; and $b$ is a parameter related to the strength of the mutation operator.

Different from the NUM, the MPTM keeps the search strength during the whole search processes [60]. When the generation increases, however, its fitness of elitisms increases accordingly. Hence it is difficult for the MPTM to spot a new individual to beat current elitisms as the population evolves to a certain extent. For example, given a point $x = (x_1, x_2, ..., x_n)$, the MPTM formulates a mutated point $\hat{x} = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)$ by

$$\hat{x}_i = (1 - \hat{t}_i)x_i^l + \hat{t}_i x_i^u \tag{6-7}$$

where $x_i^l$ and $x_i^u$ are the respective lower and upper bounds of the $i$th decision variable; $\hat{t}$ is represented as

$$\hat{t}_i = \begin{cases} t_i - t_i(\dfrac{t_i - r}{t_i})^b & \text{if } r < t_i \\ t_i & \text{if } r = t_i \\ t_i + (1-t_i)(\dfrac{r - t_i}{1-t_i})^b & \text{if } r > t_i \end{cases} \tag{6-8}$$

where $r$ is a random number uniformly distributed over [0,1], and $t_i = \dfrac{x_i - x_i^l}{x_i^u - x_i}$.

The proposed EMM operator in this work aims to maintain superior search ability in the first few generations to improve global search capability as MPTM, and to evolve the whole search space as the number of generation increases like NUM so as to fully explore the local space. If $x = (x_1, x_2, ..., x_n)$ and $\hat{x} = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)$ are the selected respective parent point and mutated point, the operation of the proposed EMM operator will be

$$\tilde{x}_i = (1 - \hat{t}_i)x_i^l + \hat{t}_i x_i^u \tag{6-9}$$

$$\hat{t}_i = \begin{cases} t_i - t_i(\dfrac{t_i - r_i}{t_i})^b & \text{if } r_i < t_i \\ t_i & \text{if } r_i = t_i \\ t_i + (1-t_i)(\dfrac{r_i - t_i}{1-t_i})^b & \text{if } r_i > t_i \end{cases} \tag{6-10}$$

$$\hat{x}_t = x_t + (\tilde{x}_t - x_t)(1 - p * v^{(1-\frac{t}{T})}) \tag{6-11}$$

$$t_i = \frac{x_i - x_i^l}{x_i^u - x_i} \tag{6-12}$$

where $r_i$ is a random number over $[0,1]$; $x_i^l$ and $x_i^u$ are the respective lower and upper bounds of the *i*th decision variable; $v \in [0,1]$ is a uniformly distributed random number; $T$ is the maximum number of generation; $b$ and $p$ determine the strength of algorithm and evolving speed of the search space, respectively.

# Chapter 7 Performance Evaluation of EGA Technique

## 7.1 Overview

To verify the effectiveness of the developed EGA technique, a series of tests will be conducted in this section based on some commonly used benchmark problems in optimization. The performance of the EGA technique will be compared with other related methods as summarized as follows:

**BC operator:**

- Scheme-1: the GA with the LC operator and the MPTM operator;
- Scheme-2: the GA with the HC operator and the MPTM operator;
- Scheme-3: the GA with the proposed BC operator and MPTM operator;

**EMM operator:**

- Scheme-4: the GA with the LC operator and the NUM operator;
- Scheme-5: the GA with the LC operator and the MPTM operator;
- Scheme-6: the GA with the LC operator and the proposed EMM operator;

**Comprehensive operation:**

- Scheme-7: GA with the LC and the MPTM operators;

- Scheme-8: EGA with the BC operator and the EMM operator.

The tests will be conducted based on the following 20 commonly used benchmark test functions:

Problem #1: the Goldstein & Price function:

$$\min f(x) = (1+(x_1+x_2+1)^2(19-14x_1+13x_1^2-14x_2+6x_1x_2+3x_2^2)$$

$$*(30+(2x_1-3x_2)^2(18-32x_1+12x_1-48x_2-36x_1x_2+27x_2^2)),$$

Conditions: $-2 \le x_j \le 2, j=1,2$, $x^* = (0,1), f(x^*) = 3$.

Problem #2: the Branin function:

$$\min f(x) = (x_2 - \frac{5}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1-\frac{1}{8\pi})\cos(x_1) + 10.$$

Conditions: $-5 \le x_1 \le 10,\ 0 \le x_2 \le 15$,

$$x^* = (-\pi,12.257),(\pi,2.275),(9.42478,2.475), f(x^*) = 0.397887.$$

Problem #3: the Beale function:

$$\min f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2.$$

Conditions: $-4.5 \le x_j \le 4.5, j=1,2, x^* = (3,0.5), f(x^*) = 0$.

Problem #4: the Michalewics function:

$$\min f(x) = -\sum_{i=1}^{2} \sin(x_i) \sin(i x_i^2 / \pi)^{2m}; m = 10 .$$

Conditions: $0 \le x_j \le \pi, j = 1,2...\text{n}; \text{n} = 2$, $f(x^*) = -1.8013$.

Problem #5: the Perm function:

$$\min f(x) = \sum_{k=1}^{n} \left[ \sum_{i=1}^{n} (i^k + \beta)(x_i / i)^k - 1 \right]^2 .$$

Conditions: $-2 \le x_j \le 2, j = 1,2...\text{n}; \text{n} = 2$, $x^* = (1,2...n), f(x^*) = 0$.

Problem #6: the Rastrigin function:

$$\min f(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i)) .$$

Conditions: $-5.12 \le x_j \le 5.12, j = 1,2...\text{n}; \text{n} = 2$, $x^* = (0,...,0), f(x^*) = 0$.

Problem #7: the Ackley function:

$$\min f(x) = 20 + e - 20e^{-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{-\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)} .$$

Conditions: $-15 \le x_j \le 30, j = 1,2...\text{n}; \text{n} = 2$, $x^* = (0,...,0), f(x^*) = 0$.

Problem #8: the Shubert function:

$$\min f(x) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1 + i)\right)\left(\sum_{i=1}^{5} i\cos((i+1)x_2 + i)\right),$$

Conditions: $-10 \le x_j \le 10, j = 1,2$, $f(x^*) = -186.7309$.

Problem #9: the Schwefel function:

$$\min f(x) = 418.9829n - \sum_{i=1}^{n}(x_i \sin\sqrt{|x_i|}),$$

Conditions: $-500 \le x_j \le 500, j = 1,2,3,$

$$x^* = (420.9867,420.9867,420.9867), f(x^*) = 0.$$

Problem #10: the Levy function:

$$\min f(x) = \sin^2(\pi y_1) + \sum_{i=1}^{n-1}\left[(y_i - 1)^2(1 + 10\sin^2(\pi y_i + 1))\right] + (y_n - 1)^2(1 + 10\sin^2(2\pi y_n)),$$

Conditions: $y_i = 1 + \frac{x_i - 1}{4}$, $-10 \le x_j \le 10, j = 1,2,3, x^* = (1,1,1), f(x^*) = 0.$

Problem #11: the Shekel function:

$$\min_x f(x) = -\sum_{j=1}^{m}\left[\sum_{i=1}^{4}(x_i - C_{ij})^2 + \beta_j\right]^{-1}, \quad \beta = \frac{1}{10}[1,2,2,4,4,6,3,7,5,5]^T, m = 10$$

$$C = \begin{bmatrix} 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7 \\ 4 & 1 & 8 & 6 & 7 & 9 & 5 & 1 & 2 & 3 \\ 4 & 1 & 8 & 6 & 3 & 2 & 3 & 8 & 6 & 7 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3 \end{bmatrix}.$$

Conditions: $0 \le x_j \le 10, j = 1,2,3,4, x^* = (4,4,4,4), f(x^*) = -10.5364.$

Problem #12: the Push-cart system:

$$x_1(k+1) = x_2(k), \quad k = 1,2,...,N,$$

$$x_2(k+1) = 2x_2(k) - x_1(k) + \frac{1}{N^2}u(k), \; x_1 = 0, x_2 = 0, N = 20,$$

$$\min_u f(x,u) = -x_1(N+1) + \frac{1}{2N}\sum_{k=1}^{N}u(k)^2 \; . \; f(u^*) = -\frac{1}{3} + \frac{3N-1}{6N^2} + \frac{1}{2N^3}\sum_{k=1}^{N-1}k^2,$$

Conditions: $u^* = (0.95,0.9,0.85,0.8,0.75,0.7,0.65,0.6,0.55,$

$$0.5,0.45,0.4,0.35,0.3,0.25,0.2,0.15,0.1,0.05,0).$$

Problem #13: the Hartmann (H3,4) function:

$$\min f(x) = -\sum_{i=1}^{4} \alpha_i \exp\left[ -\sum_{j=1}^{3} B_{ij} (x_j - Q_{ij})^2 \right], \alpha = [1,1.2,3,3.2]^T,$$

$$B = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}, \; Q = 10^{-4} \begin{bmatrix} 6890 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{bmatrix}.$$

Conditions: $0 < x_j < 1, j = 1,2,3$,

$$x^* = (0.114614, 0.555469, 0.852547), f(x^*) = -3.86278.$$

Problem #14: the Zakharov function:

$$\min f(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5 i x_i)^2 + (\sum_{i=1}^{n} 0.5 i x_i)^4,$$

Conditions: $-5 \le x_j \le 10, j = 1,2...\text{n}; \text{n} = 4, \; x^* = (0,...,0), f(x^*) = 0$.

Problem #15: the Trid function:

$$\min f(x) = \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1},$$

Conditions: $-n^2 \le x_j \le n^2, j = 1,2...\text{n}; \text{n} = 10, \; f(x^*) = -200$.

Problem #16: the Sum Squares function:

$$\min f(x) = \sum_{i=1}^{n} i x_i^2,$$

Conditions: $-10 \le x_j \le 10, j = 1,2...\text{n}; \text{n} = 10, \; x^* = (0,...,0), f(x^*) = 0$.

Problem #17: the Sphere function:

$$\min f(x) = \sum_{i=1}^{n} x_i^2,$$

Conditions: $-5.12 \le x_j \le 5.12, j = 1,2...\text{n}; \text{n} = 20, \; x^* = (0,...,0), f(x^*) = 0$.

Problem #18: the Rosenbrock function:

$$\min f(x) = \sum_{i=1}^{n-1} \left[ 100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2 \right],$$

Conditions: $-5 \le x_j \le 10, j = 1,2,3, x^* = (1,1,1), f(x^*) = 0$.

Problem #19: the Powell function:

$$\min f(x) = \sum_{i=1}^{n/4} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} + 10x_{4i})^2 + (x_{4i-2} + 10x_{4i-1})^4 + 10(x_{4i-3} + 10x_{4i})^4,$$

Conditions: $-4 \le x_j \le 5, j = 1,2,3,4, x^* = (3,-1,0,1), f(x^*) = 0$.

Problem #20: the Dixon & Price function:

$$\min f(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$$

Conditions: $-10 \le x_j \le 10, j = 1,2,3,4,5, f(x^*) = 0$.

The simulation tests will be terminated as long as the error becomes less than Ed = {$10^{-10}$, $10^{-6}$, $10^{-9}$, $10^{-4}$, $10^{-5}$, $10^{-10}$, $10^{-6}$, $10^{-7}$, $10^{-4}$, $10^{-6}$, $10^{-4}$, $10^{-6}$, $10^{-5}$, $10^{-6}$, $10^{-5}$, $10^{-5}$, $10^{-6}$, $10^{-5}$, $10^{-4}$, $10^{-6}$} correspondingly to problems #1-#20, respectively, or the maximum generation number is achieved ($10^4$ in this case).

## 7.2 Comparison of Crossover Operators

The first test is to verify the effectiveness of the proposed BC operator (Scheme-3). The comparison is with the classical LC (Scheme-1) and HC (Scheme-2) operators. All the crossover operators are combined with the MPTM for mutation operations. The number of population is given as 20. The number of elitism kids, crossover kids and mutation kids are set as 2, 13 and 5,

respectively. The Roulette Wheel selection is applied in all the tests to ensure that offspring are more likely derived from the parents with higher fitness.

The detailed test results are summarized in Table 7.1. Figure 7.1 shows the averaged test results of Scheme-1 to Scheme-3 over 30 tests, corresponding to different crossover operators (LC, HC, and the proposed BC operators). From Figure 7.1(a), it is seen that Scheme-3 provides the best success rates for these test problems. Although Scheme-3 could not reach 100% success rate for Problem #11 under the employed termination criteria, it still performs better than Schemes-1 and 2.

By examining Figures 7.1(b) and 7.1(c), it is clear that the developed BC operator (Scheme-3) outperforms the classical HC and LC operators in terms of number of function evaluation (Figure 7.1b) and execution time (Figure 7.1c). The proposed BC operator can explore the local space more thoroughly than other crossover operators. It takes much fewer generations to reach minima than other classical crossover operators; and thus its overall search speed is improved significantly.

On the other hand, from Figure 7.1(b) and Figure 7.1(c), it is seen that there is no big difference in the performance of HC operator (Scheme-2) and the LC operator (Scheme-1) in terms of execution time and number of evaluations. It means that each of these two classical crossover operators has its own advantages and disadvantages in optimization applications.

Table 7.1: Comparison results of Scheme-1 to Scheme-3 corresponding to different crossover operators

| Problem number | Successful rate (%) (out of 30 runs) | | | Average number of function evaluations | | | Average execution time (s) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Scheme1 | Scheme2 | Scheme3 | Scheme1 | Scheme2 | Scheme3 | Scheme1 | Scheme2 | Scheme3 |
| P1 | 100 | 100 | 100 | 13800 | 7080 | 4400 | 0.694 | 0.337 | 0.307 |
| P2 | 100 | 100 | 100 | 5520 | 7040 | 940 | 0.262 | 0.317 | 0.060 |
| P3 | 100 | 100 | 100 | 9454 | 14860 | 6411 | 0.506 | 0.931 | 0.463 |
| P4 | 100 | 100 | 100 | 3600 | 6480 | 840 | 0.190 | 0.317 | 0.058 |
| P5 | 100 | 100 | 100 | 8300 | 16240 | 1260 | 0.445 | 0.812 | 0.087 |
| P6 | 100 | 100 | 100 | 22920 | 17980 | 13328 | 1.170 | 0.864 | 0.831 |
| P7 | 100 | 100 | 100 | 14620 | 19880 | 4680 | 0.849 | 1.082 | 0.350 |
| P8 | 100 | 100 | 100 | 7739 | 27134 | 1627 | 0.391 | 1.287 | 0.105 |
| P9 | 100 | 100 | 100 | 29298 | 33010 | 15401 | 1.631 | 1.896 | 1.079 |
| P10 | 100 | 100 | 100 | 19874 | 25705 | 3011 | 1.173 | 1.459 | 0.218 |
| P11 | 83.3 | 63.3 | 86.6 | 87932 | 133029 | 45496 | 6.709 | 9.780 | 4.187 |
| P12 | 100 | 100 | 100 | 51300 | 164434 | 19275 | 3.759 | 11.636 | 1.648 |
| P13 | 100 | 100 | 100 | 10236 | 9010 | 2872 | 0.775 | 0.659 | 0.268 |
| P14 | 80 | 100 | 100 | 50581 | 24579 | 3779 | 2.608 | 1.172 | 0.258 |
| P15 | 100 | 100 | 100 | 78125 | 113045 | 18253 | 4.005 | 5.482 | 1.167 |
| P16 | 100 | 100 | 100 | 103177 | 48778 | 15872 | 5.078 | 2.266 | 1.026 |
| P17 | 100 | 100 | 100 | 144396 | 126269 | 21906 | 7.879 | 6.280 | 1.441 |
| P18 | 83.3 | 83.3 | 100 | 74570 | 118867 | 29533 | 3.589 | 5.438 | 1.854 |
| P19 | 100 | 100 | 100 | 45616 | 32956 | 4742 | 3.004 | 2.053 | 0.381 |
| P20 | 86.6 | 76.6 | 100 | 67539 | 41130 | 27222 | 3.492 | 1.994 | 1.738 |

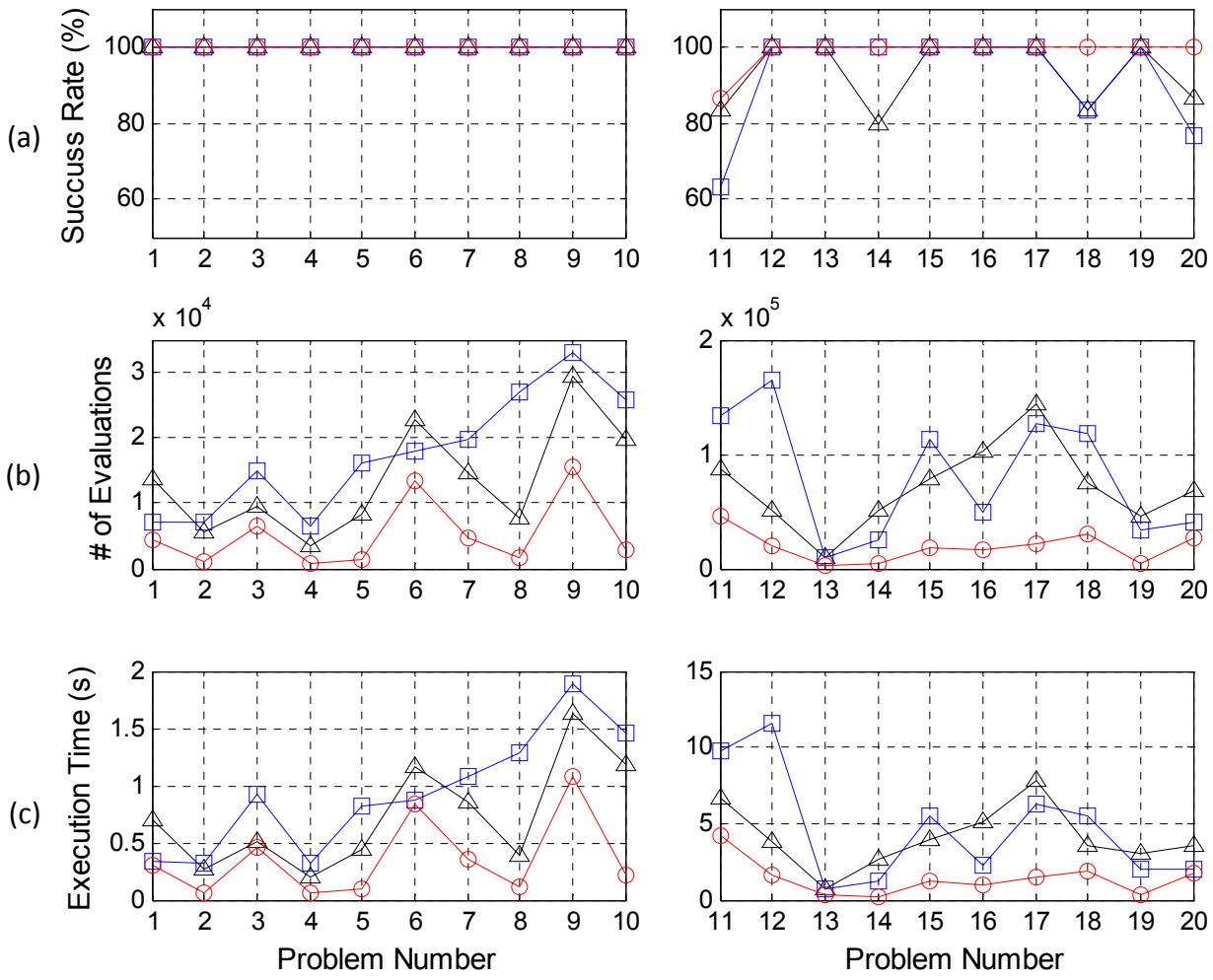


Figure 7.1. Performance comparison of Scheme-1 (triangle line), Scheme-2 (square line) and Scheme-3 (circle line) in terms of: (a) Successful rate, (b) Number of function evaluation, and (c) Execution time.

## 7.3 Mutation Comparison

The tests in this subsection are to verify the effectiveness of the proposed EMM mutation operator. The LC operator is utilized for crossover operations in all three schemes. Figure 7.2 illustrates the averaged test results using the proposed EMM mutation operator (Scheme-6) compared with the classical operators of NUM (Scheme-4) and MPTM (Scheme-5); detailed results are summarized in Table 7.2. It is seen from Figure 7.2(a) that the suggested EMM

method provides the highest successful rate compared with the classical NUM (Scheme-4) and MPTM (Scheme-5) mutation operators under the applied termination criteria. Based on the successful rate, its advantage becomes even more apparent for problems with complex search spaces such as Problems #11, #14, #18, and #20.

From Figure 7.2(b) and Figure 7.2(c), it is seen that the classical NUM (Scheme-4) and MPTM (Scheme-5) operators each has their own advantage and shortcomings of overall performance and execution time according to different test problems. However, the suggested EMM operator (Scheme-6) provides best performance by taking not only the least number of function evaluations (Figure 7.2b) but also the fastest operations (Figure 7.2c). The EMM operator can expand the species diversity so as to enhance the global search capability of the EGA technique by preventing the possible trapping due to local minima.

Table 7.2: Comparison results of Scheme-4 to Scheme-6 corresponding to different mutation operators

| Problem number | Successful rate (%) (out of 30 runs) | | | Average number of function evaluations | | | Average execution time (s) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Scheme 4 | Scheme5 | Scheme6 | Scheme4 | Scheme5 | Scheme6 | Scheme4 | Scheme5 | Scheme6 |
| P1 | 100 | 100 | 100 | 12600 | 13800 | 11960 | 0.637 | 0.694 | 0.591 |
| P2 | 100 | 100 | 100 | 6420 | 5520 | 4140 | 0.309 | 0.262 | 0.203 |
| P3 | 100 | 100 | 100 | 9819 | 9454 | 7214 | 0.558 | 0.506 | 0.392 |
| P4 | 100 | 100 | 100 | 4000 | 3600 | 3160 | 0.212 | 0.190 | 0.171 |
| P5 | 100 | 100 | 100 | 9140 | 8300 | 6780 | 0.494 | 0.445 | 0.367 |
| P6 | 100 | 100 | 100 | 27140 | 22920 | 14800 | 1.398 | 1.170 | 0.785 |
| P7 | 100 | 100 | 100 | 21800 | 14620 | 11780 | 1.259 | 0.849 | 0.624 |
| P8 | 100 | 100 | 100 | 8766 | 7739 | 7110 | 0.452 | 0.391 | 0.370 |
| P9 | 100 | 100 | 100 | 24360 | 29298 | 21456 | 1.466 | 1.631 | 1.228 |
| P10 | 100 | 100 | 100 | 36438 | 19874 | 15038 | 2.190 | 1.173 | 0.907 |
| P11 | 76.6 | 83.3 | 100 | 96333 | 87932 | 30460 | 7.366 | 6.709 | 2.693 |
| P12 | 100 | 100 | 100 | 59980 | 51300 | 45080 | 4.618 | 3.759 | 3.616 |
| P13 | 100 | 100 | 100 | 16474 | 10236 | 9403 | 1.254 | 0.775 | 0.722 |
| P14 | 80 | 80 | 96.6 | 60350 | 50581 | 40322 | 3.208 | 2.608 | 2.203 |
| P15 | 100 | 100 | 100 | 74109 | 78125 | 65842 | 3.928 | 4.005 | 3.594 |
| P16 | 100 | 100 | 100 | 117688 | 103177 | 70407 | 6.070 | 5.078 | 3.754 |
| P17 | 100 | 100 | 100 | 110466 | 144396 | 83609 | 6.134 | 7.879 | 4.900 |
| P18 | 66.6 | 83.3 | 90 | 102199 | 74570 | 58526 | 5.003 | 3.589 | 3.003 |
| P19 | 100 | 100 | 100 | 58582 | 45616 | 37058 | 4.263 | 3.004 | 2.551 |
| P20 | 93.3 | 86.6 | 100 | 76759 | 67539 | 64877 | 3.794 | 3.492 | 3.235 |

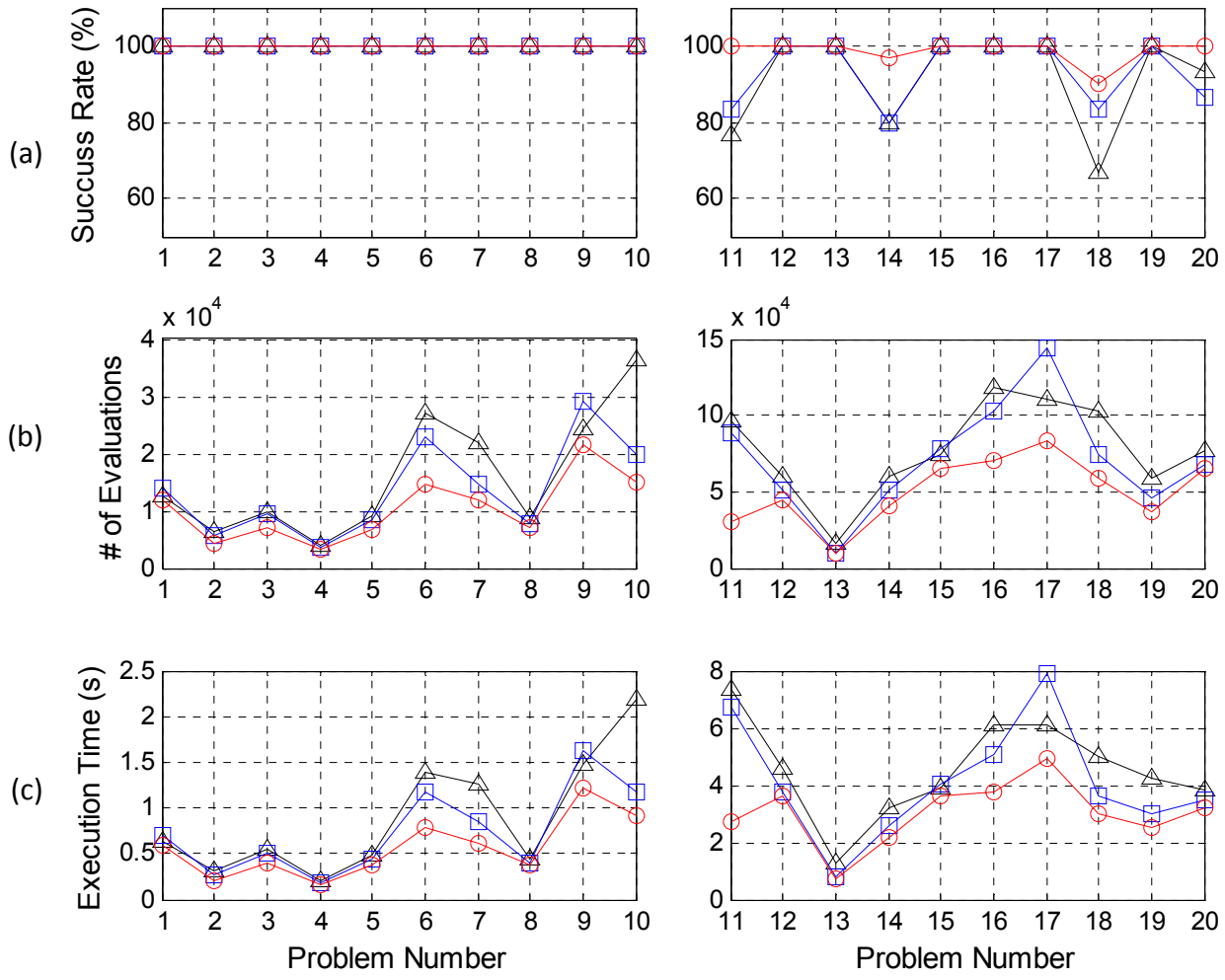


Figure 7.2. The comparison of Scheme-4 (triangle line), Scheme-5 (square line) and Scheme-6 (circle line) in terms of: (a) Successful rate, (b) Number of function evaluation, and (c) Execution time.

## 7.4 Comprehensive Comparison

More tests are taken in this subsection to verify the effectiveness of the proposed EGA technique with both BC and EMM operators (Scheme-8). Its performance is compared with a classic GA using the LC and MPTM methods (Scheme-7). Figure 7.3 demonstrates the comparison results using the same benchmark test functions; detailed results are summarized in Table 7.3. It is clear that Scheme-8 outperforms the classical GA method in terms of the

successful rate, number of function evaluations and execution time. EGA is the only scheme that can achieve 100% of successful rates for all of these tested examples (Figure 7.3a); it is even superior to Scheme-3 using BC and MPTM operators (Figure 7.1a) and Scheme-6 using EMM and LC operators (Figure 7.2a), especially for problems with complex search spaces such as problems #11, #14 and #18. Therefore, it can be concluded that the developed EGA technique can effectively improve convergence speed, successful rate and global search capability of the classical GA methods.

Table 7.3: Comparison results of Scheme-7 and Scheme-8 corresponding to GA and EGA

| Problem number | Successful rate (%) (out of 30 runs) | | Average number of function evaluations | | Average execution time (s) | |
|---|---|---|---|---|---|---|
| | Scheme7 | Scheme8 | Scheme7 | Scheme8 | Scheme7 | Scheme8 |
| P1 | 100 | 100 | 13800 | 2840 | 0.694 | 0.192 |
| P2 | 100 | 100 | 5520 | 580 | 0.262 | 0.042 |
| P3 | 100 | 100 | 9454 | 5460 | 0.506 | 0.349 |
| P4 | 100 | 100 | 3600 | 760 | 0.190 | 0.056 |
| P5 | 100 | 100 | 8300 | 700 | 0.445 | 0.053 |
| P6 | 100 | 100 | 22920 | 6920 | 1.170 | 0.480 |
| P7 | 100 | 100 | 14620 | 3440 | 0.849 | 0.239 |
| P8 | 100 | 100 | 7739 | 1440 | 0.391 | 0.097 |
| P9 | 100 | 100 | 29298 | 12270 | 1.631 | 0.827 |
| P10 | 100 | 100 | 19874 | 2670 | 1.173 | 0.197 |
| P11 | 83.3 | 100 | 87930 | 20960 | 6.709 | 1.955 |
| P12 | 100 | 100 | 51300 | 18522 | 3.759 | 1.639 |
| P13 | 100 | 100 | 10236 | 2202 | 0.775 | 0.211 |
| P14 | 80 | 100 | 50581 | 2926 | 2.608 | 0.190 |
| P15 | 100 | 100 | 78125 | 15376 | 4.005 | 1.046 |
| P16 | 100 | 100 | 103177 | 14816 | 5.078 | 0.995 |
| P17 | 100 | 100 | 144396 | 17142 | 7.879 | 1.198 |
| P18 | 83.3 | 100 | 74570 | 21514 | 3.589 | 1.320 |
| P19 | 100 | 100 | 45616 | 4068 | 3.004 | 0.334 |
| P20 | 86.6 | 100 | 67539 | 22892 | 3.492 | 1.420 |

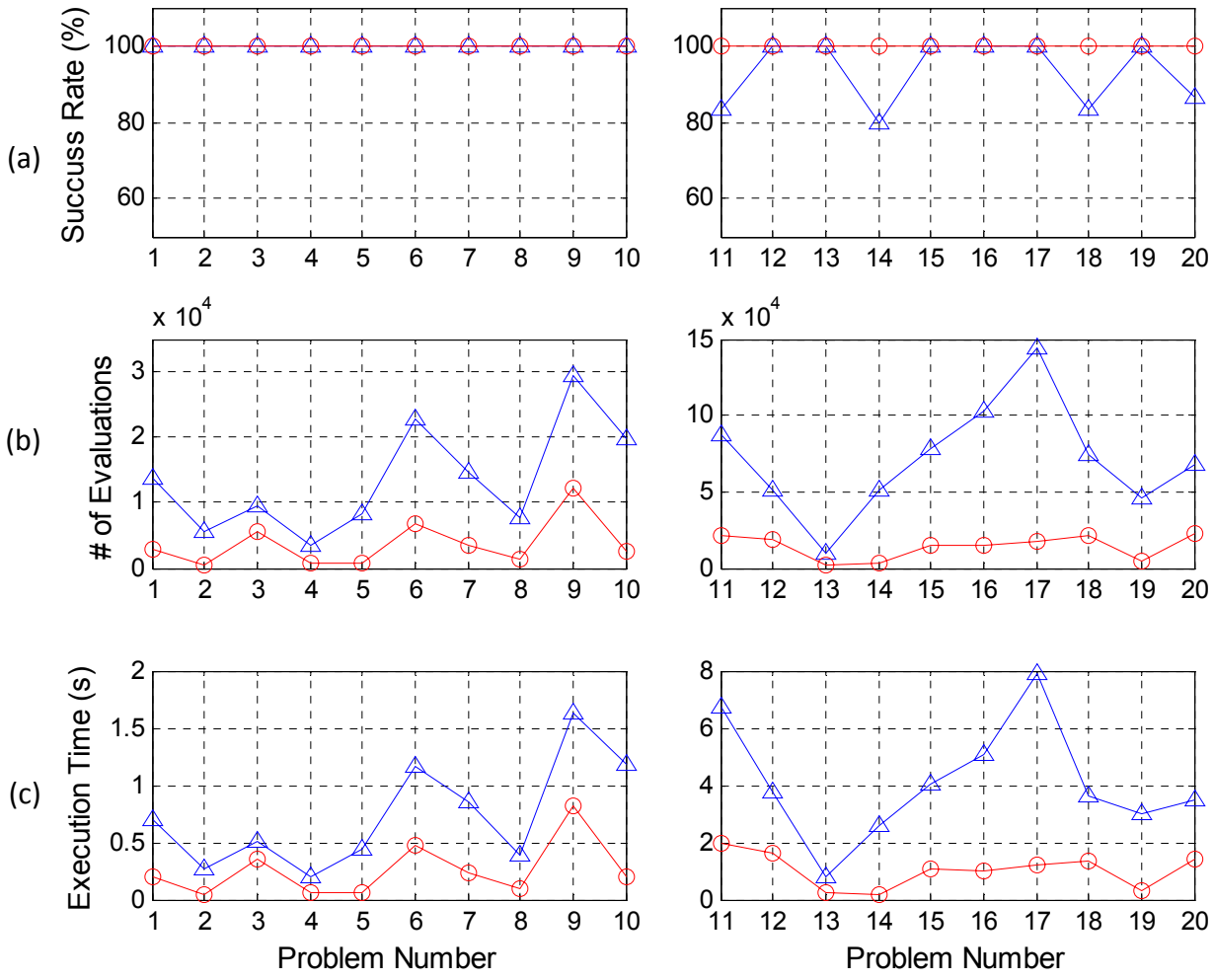


Figure 7.3. The comparison of Scheme-7 (triangle line) and Scheme-8 (circle line) in terms of: (a) Successful rate, (b) Number of function evaluation, and (c) Execution time.

# Chapter 8 Conclusions and Future Work

## 8.1 Conclusions

In this thesis, an enhanced sliding mode (ESM) control strategy is developed for vibration suppression in flexible structures. By incorporating *T(.)* function instead of terminal attractor, it can achieve the ability of terminal attractor while prevent certain implementation problems. An NF approximator is suggested to identify the system dynamics. Two new training techniques, EG technique and modified LSE, are proposed to train the nonlinear parameters and linear parameters of the NF system, respectively. The hybrid training technique based on the novel learning methods can not only generate fast convergence and global stability but also improve control performance. The effectiveness of the proposed controller and training techniques are verified on the flexible structure workstation with nonlinear system properties. Test results have shown that the developed ESM controller outperforms other related control strategies in term of settling time and overshoot.

Another contribution of this thesis is the development of a novel extended GA (or EGA) technique for global system training and optimization. A novel BC operator is proposed to speed up the search process and expand the local search space. An EMM operator is suggested to prevent possible trapping due to local minima. The effectiveness of the developed EGA technique has been verified by a series of simulation tests corresponding to 20 benchmark optimization examples. Test results have shown that both the BC and EMM operators outperform their classical counterparts in terms of execution time and successful rate. EGA with BC and EMM methods can provide superior performance over the classic GA methods.

## 8.2 Future Works

1) A more efficient training technique will be developed and implemented to facilitate system identification so as to further improve control performance, especially for undershoot.

2) The proposed EGA technique will be further tested, and then implemented for real system training and optimization.

3) The intelligent sliding mode controller can be improved and adjusted for other flexible systems such as 2-DOF robots with flexible arms.

4) A more efficient machine learning technique such as least square support vector machine will be developed for system training and optimization.

# References


[1] I. N. Kar, K. Seto, F. Doi, "Multimode vibration control of a flexible structure using H∞-based robust control," IEEE/ASME Transactions on Mechatronic, vol. 5, no. 1, pp. 23-31, 2000.

[2] S. Hashimoto, K. Hara, H. Funato, K. Kamiyama, "AR-based identification and control approach in vibration suppression," IEEE Transactions on Industrial Applications, vol. 37, no. 3, pp. 806-811, 2001.

[3] Y. Izumikawa, K. Yubai, J. Hirai, "Fault-tolerant control system of flexible arm for sensor fault by using reaction force observer," IEEE/ASME Transactions on Mechatronic, vol. 10, no. 4, pp. 391-396, 2005.

[4] K. Takagi, H. Nishimura, "Control of a jib-type crane mounted on a flexible structure," IEEE Transactions on Control Systems Technology, vol. 11, no. 1, pp. 32-42, 2003.

[5] D. H. S. Maithripala, J. M. Berg, and W. P. Dayawansa, "Control of an electrostatic MEMS using static and dynamic output feedback," ASME Journal of Dynamic Systems, Measurement, and Control, vol. 127, no. 3, pp. 443–450, 2005.

[6] R.-E. Precup, S. Preitl, and P. Korondi, "Fuzzy controllers with maximum sensitivity for servosystems," IEEE Transactions on Industrial Electronics, vol. 54, no. 3, pp. 1298-1310, 2007.

[7] M. Ishikawa, Y. Minami, and T. Sugie, "Development and control experiment of the trident snake robot," IEEE/ASME Transactions on Mechatronics, vol. 15, no. 1, pp. 9-16, 2010.

[8] K.D. Young, U. Ozguner, Frequency shaping compensator design for sliding mode, International Journal of Control, vol. 57, pp. 1005-1019, 1993.

[9] Y. Kitamura, K. Iwabuchi, K. Nonami, H. Nishimura, Positioning control of flexible arm using frequency-shaped sliding mode control, in: Third International Conference on Motion and Vibration Control, pp. 178-183, 1996.

[10] K. Nonami, H. Nishimura, H. Tian, H∞/μcontrol-based frequency shaped sliding mode control for flexible structures, JSME International Journal, 1996.

[11] C. Pai, and A. Sinha, Sliding Mode Output Feedback Control of Vibration in a Flexible Structure, Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control, vol. 129, pp. 851- 855, 2007.

[12] G. Song, and H. Gu, Active Vibration Suppression of a Smart Flexible Beam Using a Sliding Mode Based Controller, Journal of Vibration and Control, vol. 13, pp. 1095–1107, 2007.

[13] M. Itik, M. U. Salamci, F. D. Ulker, and Y. Yaman. Active Vibration Suppression of a Flexible Beam via Sliding Mode and Hinf Control. Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, pp. 12-15, 2005.

[14] H. C. Gu and G. B. Song. Adaptive robust sliding-mode control of a flexible beam using PZT sensor and actuator. Proceedings of the IEEE International Symposium on Intelligent Control, pp. 78-83, 2004.

[15] M. C. Pai, and A. Sinha. Sliding Mode Output Feedback Control of Vibration in a Flexible Structure. Journal of Dynamic Systems, Measurement, and Control. vol. 129, pp. 851-855, 2007.

[16] Mark W. Spong, M. Vidyasagar, Robot Dynamics and Control, Toronto: Wiley, 1989.

[17] S. Huang, K. Huang, "An adaptive fuzzy sliding mode controller for servomechanism disturbance rejection," IEEE Transactions on Industrial Electronics, vol. 48, no. 4, pp. 845-852, 2001.

[18] Q. Hu, Z. Wang, H. Gao, "Sliding mode and shaped input vibration control of flexible systems," IEEE Transactions on Aerospace and Electronic systems, vol. 44, no. 2, pp. 503-519, 2008.

[19] T. C. Manjunath, B. Bandyopadhyay, "Vibration control of timoshenko smart structures using multirate output feedback based discrete sliding mode control for SISO systems," Journal of Sound and Vibration, vol. 326, pp. 50-74, 2009.

[20] G. Song, H. Gu, "Active vibration suppression of a smart flexible beam using a sliding mode based controller," Journal of Vibration and Control, vol. 13, no. 8, pp. 1095-1107, 2007.

[21] S. Choi, Y. Han, "Vibration control of electrorheological seat suspension with human-body model using sliding mode control," Journal of Sound and Vibration, vol. 303, pp. 391-404, 2007.

[22] X. Xue, J. Tang, "Robust and high precision control using piezoelectric actuator circuit and integral continuous sliding mode control design," Journal of sound and vibration, vol. 293, pp. 335-359, 2006.

[23] Z. Man, A. P. Paplinski, H. Wu. "A robust MIMO terminal sliding mode control scheme for rigid robot manipulators." IEEE Transactions on Automatic Control, vol. 39, pp. 2464 – 2469, 1994.

[24] K. B. Park, J. Lee. “Comments on ‘A robust MIMO terminal sliding mode control scheme for rigid robot manipulators.” IEEE Transactions on Automatic Control, vol. 41, no. 5, pp. 761 – 762, 1996.

[25] Y. Feng, X. Yu, Z. Man. “Non-singlar terminal sliding mode control of rigid robot manipulators.” Automatica, vol. 38, no. 12, pp. 2159 – 2167, 2002.

[26] K. B. Park, T. Tsuiji. “Terminal sliding mode control of second-order nonlinear uncertain systems.” International Journal of Robust and Nonlinear Control, vol. 9, no. 11, pp. 769 – 780, 1999.

[27] K. Y. Zhuang, H. Y. Su, K. Q. Zhuang, J. Chu. “Adaptive terminal sliding mode control for high order nonlinear dynamic systems.” Journal of Zhejiang University, vol. 4, pp. 58 – 63, 2003.

[28] R. JAMIL, “Inverse dynamics based tuning of a fuzzy logic controller for a single-link flexible manipulator.” Journal of Vibration and Control, vol. 13, pp. 1741–1759, 2007.

[29] G. Yan, and Lily L. Zhou. “Fuzzy logic and genetic algorithms for intelligent control of structures using MR dampers.” Proc. of SPIE conference on Smart Structures and Materials: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems, vol. 5391, pp. 555-565, 2004.

[30] II. Kim, J. H. Lee, and E. O. Bang. “A new approach to adaptive membership function for fuzzy inference system.” Third International Conference on Knowledge-Based Intelligent Information Engineering Systems, Adelaide, Australia pp. 112-116, 1999.

[31] A. Jnifene, and W. Andrews. “Fuzzy logic control of the end-point vibration in an experimental flexible beam.” Journal of Vibration and Control, vol. 10, pp. 493-506, 2004.

[32] T. H. S. Li and S. H. Tsai. “T-S fuzzy bilinear model and fuzzy controller design for a class of nonlinear systems.” IEEE Transactions on fuzzy systems, vol. 15, pp. 494-506, 2007.

[33] C. S. Ting. “An observer-based approach to controlling time-delay chaotic systems via Takagi-Sugeno fuzzy model.” Information Sciences, vol. 177, pp. 4314–4328, 2007.

[34] L. Tian, and C. Collins. “A dynamic recurrent neural network-based controller for a rigid-flexible manipulator system”. Mechatronics, vol. 14, pp. 471–490, 2004.

[35] S. Bruckner, and S. Rudolph. “Neural networks applied to smart structure control.” Proceedings SPIE Conference on Applications and Science of Computational Intelligence III, April 2000.

[36] J. Harris. “Comparative aspects of neural networks and fuzzy logic for real-time control.” Neural networks for control and systems, pp. 72-93, 1992.

[37] J.-S. R. Jang, C.-T. Sun, E. Mizutani, Neuro-fuzzy And Soft Computing, New Jersey: Prentice Hall, 1997.

[38] L. Wang and J. Mendel, “Fuzzy basis functions, universal approximation, and orthogonal least squares learning,” IEEE Transaction on Neural Networks, vol. 3, no. 5, pp. 807-814, 1992.

[39] S. Q. Wu, M. J. Er, “Dynamic fuzzy neural networks-a novel approach to function approximation,” IEEE Transaction on Systems Man Cybernetics-Part B, vol. 30, no. 2, pp. 358-364, 2000.

[40] S. Chen, C. F. N. Cowan, P. M. Grant, “Orthogonal least squares learning algorithm for radial basis function networks,” IEEE Transaction on Neural Networks, vol. 2, no. 2, pp. 302-309, 1991.

[41] N. Qian, “On the momentum term in gradient descent learning algorithms,” Neural Networks, vol. 12, no. 1, pp. 145-151, 1999.

[42] S. I. Hill, R. C. Williamson, “Convergence of exponentiated gradient algorithm,” IEEE Transaction on Signal Process, vol. 49, no. 6, pp. 1208-1215, 2001.

[43] Shuanghe Yu, Xinghuo Yu, Zhihong Man, “A fuzzy neural network approximator with fast terminal sliding mode and its applications,” Fuzzy Sets and Systems, vol. 148, no. 3, pp. 469-486, 2004.

[44] S. Venkatraman, G. G. Yen, “A generic framework for constrained optimization using genetic algorithms,” IEEE Transactions on Evolutionary Computation, vol. 9, no. 4, pp. 424-435, 2005.

[45] R. Kumar, K. Izui, Y. masataka, S. Nishiwaki, “Multilevel redundancy allocation optimization using hierarchical genetic algorithm,” IEEE Transactions on Reliability, vol. 57, no. 4, pp.650-661, 2008.

[46] J. Tsai, T. Liu, J. Chou, “Hybrid taguchi-genetic algorithm for global, numerical optimization” IEEE Transactions on Evolutionary Computation, vol. 8, no. 4, pp. 365-377, 2004.

[47] W. Wang and D. Kanneg, “An integrated classifier for condition monitoring in transmission systems,” Mechanical Systems and Signal Processing, vol. 23, no. 4, pp. 1298-1312, 2009.

[48] C. Wang, H. Liu, C. Lin, “Dynamic optimal learning rates of a certain class of fuzzy neural networks and its applications with genetic algorithm,” IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, vol. 31, no. 3, pp. 467-475, 2001.

[49] W. Wang, Y. Li, "Evolutionary learning of BMF fuzzy-neural networks using a reduced-form genetic algorithm," IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, vol. 33, no. 6, pp. 966-976, 2003.

[50] W. Wang, F. Ismail, F. Golnaraghi, "A neuro-fuzzy approach to gear system monitoring," IEEE Transactions on Fuzzy Systems, vol. 12, pp. 710-723, 2004.

[51] L. Chen, "Real coded genetic algorithm optimization," Journal of the American Water Resources Association, vol. 39, no. 5, pp. 1157-1165, 2003.

[52] T. Srikanth, V. Kamala, "A real coded genetic algorithm for optimization of cutting parameters in turning," International Journal of Computer Science and Network Security, vol. 8, no. 6, pp. 189-193, 2008.

[53] K. Deep, M. Thakur, "A new crossover operator for real coded genetic algorithms," Applied mathematics and computation, vol. 188, pp. 895-911, 2007.

[54] C. García-Martínez, M. Lozano, F. Herrera, D. Molina, A.M. Sánchez, "Global and local real-coded genetic algorithms based on parent-centric crossover operators," European Journal of Operational Research, vol. 185, pp. 1088-1113, 2008.

[55] Z. H. Cui, J. C. Zeng, Y. B. Xu, "A new nonlinear genetic algorithm for numerical optimization," IEEE International Conference on Systems, Man and Cybernetics, vol. 5, pp. 4660-4663, 2003.

[56] W. Paszkowicz, "Properties of a genetic algorithm equipped with a dynamic penalty function," Computational Materials Science, vol. 45, pp. 77-83, 2009.

[57] S. F. Hwang, R. S. He, "A hybrid real-parameter genetic algorithm for function optimization," Advanced Engineering Informatics, vol. 20, pp. 7-21, 2006.

[58] O. Hrstka, A. Kučerová, "Improvements of real coded genetic algorithms based on differential operators preventing premature convergence," Advances in Engineering Software, vol. 35, pp. 237-246, 2004.

[59] W. Wang and O. Jianu, "A smart monitor for machinery fault detection," IEEE/ASME Transactions on Mechatronics, vol. 15, no. 1, pp. 70-78, 2010.

[60] R.A.E. Makinen, J. Periaux and J. Toivanen, "Multidisciplinary shape optimization in aerodynamics and electromagnetic using genetic algorithms," International Journal for Numerical Methods in Fluids, vol. 30, pp. 149–159, 1999.

[61] V. Utkin and H. Lee, "Chattering problem in sliding mode control systems," The 2nd IFAC Conference on Analysis and Design of Hybrid Systems, 2006.

[62] **D. Z. Li** and W. Wang, "An intelligent sliding mode controller for vibration suppression in flexible structures," Journal of Vibration and Control. (Submitted)

[63] **D. Z. Li** and W. Wang, "An Enhanced GA Technique for Optimization and System Training," IEEE Systems Man and Cybernetics, Part B. (Submitted)