Machine Learning Techniques for Large-Scale System Modeling

by

Jiaqing Lv

A Thesis
submitted to the Faculty of Graduate Studies of
The University of Manitoba
in Partial Fulfilment of the Requirements for the degree of

Master of Science

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba, R3T 5V6 Canada

Dedicated to my parents, with unfading love.

Abstract

This thesis is about some issues in system modeling: The first is a parsimonious representation of MISO Hammerstein system, which is by projecting the multivariate linear function into a univariate input function space. This leads to the so-called semiparamtric Hammerstein model, which overcomes the commonly known “Curse of dimensionality” for nonparametric estimation on MISO systems. The second issue discussed in this thesis is orthogonal expansion analysis on a univariate Hammerstein model and hypothesis testing for the structure of the nonlinear subsystem. The generalization of this technique can be used to test the validity for parametric assumptions of the nonlinear function in Hammersteim models. It can also be applied to approximate a general nonlinear function by a certain class of parametric function in the Hammerstein models. These techniques can also be extended to other block-oriented systems, e.g, Wiener systems, with slight modification. The third issue in this thesis is applying machine learning and system modeling techniques to transient stability studies in power engineering. The simultaneous variable section and estimation lead to a substantially reduced complexity and yet possesses a stronger prediction power than techniques known in the power engineering literature so far.

Keywords: nonparametric estimation, semiparametric, MISO Hammerstein model, curse of dimensionality, model selection, Lasso, transient stability boundary, machine learning

Acknowledgements

Many people have been a great help to me in my research and it is my pleasure to thank them all.

- First and foremost, I thank Professor Dr. Mirosław Pawlak for giving me the opportunity to study the MSc program under his supervision. It has been my privilege to be his student. I appreciate his guidance, encouragement, support, and advice.
- I thank Manitoba Hydro for the financial support.
- I also thank Professor U.D. Annakkage, and Dr. Bathiya Jayasekara for providing data for the research work in Section 5 of this thesis.
- I thank Professor. Mirosław Pawlak, Professor Pradeepa Yahampath, and Professor A. Thavaneswaran for their invaluable suggestions for modifying this thesis.
- Finally, I would like to thank Dr. Waner Su, Xiaoxia Li, Melvin & Corrine Klassen, Daniel Lockery, Dr. Piotr Wasilewski, Amy Dario, as well as all my other valued friends. Their friendship are also important in the process of my pursuit for knowledge.

Contents

List of Tables

List of Figures

1 Introduction

System identification is the field of applied science that is concerned with determining the structure of an unknown system from the observed data representing the input and output signals of the system. The input data are in the form of stochastic time series, whereas the output signal is corrupted by measurement noise. As such the system identification problem is closely related to statistical inference - the view that is taken in this thesis.

System identification plays an important role in science and engineering. A historical example of applying the idea of system modeling is the Italian physicist, mathematician, astronomer and philosopher Galileo Galilei, who based on the empirical observations of distance and time elapsed during the process when an object fell from a ramp, established the law of falling bodies.

Modern system identification has been utilizing statistical tools, e.g., machine learning techniques. The issue of finding an adequate parsimonious model from data plays a critical role. In fact, system identification provides powerful techniques for building models of complex systems in communication, signal processing, control, power engineering, and biomedical engineering [1], [2], [3], [4], [5], [6], [7].

Our main focus is to develop a basic methodology for a large-scale system modeling problems. The thesis is divided into four main parts. The first part is a brief introduction to basic structures of our methodology. Concepts such as block-oriented systems, and in particular models such as Hammerstein systems and Wiener systems will be introduced. The second part of the thesis is concerned with a low-dimensional representation of a multivariate Hammerstein system. This is achieved by using the theory of semiparametric approximation. Semiparametric models are a parsimonious compromise between fully nonparametric models and parametric models. Therefore, they overcome the well-known curse of dimensionality problem, and yet preserve their advantage of not assuming any parametric forms for nonlinear submodels. The third part is about identification methods using orthogonal expansions in determining the nonlinear characteristic of a single-input-single-output (SISO) Hammerstein system. Hypothesis testing is also discussed in this part in the context of testing whether the nonlinear function within the Hammerstein system belongs to a certain parametric class. Finally, the last part of the thesis deals with a particular machine learning techniques applications, namely, the study of transient stability analysis of power systems. Specifically, we use the regression methods utilizing Least Absolute Shrinkage and Selection Operator (Lasso), compared with the other methods used in the field of power engineering. The contributions presented in this thesis are:

- Numerical studies of semiparametric approximation for MISO Hammerstein models (Section 3),
- Using the thresholding approach to determine the structure of orthogonal basis estimates for the nonlinear characteristics of a SISO Hammerstein system, where it is assumed that the nonlinear subsystem belongs to a given parametric class (Section 4),
- Truncation parameter selection for orthogonal series estimate of the Hammerstein system (Section 4),
- Testing hypothesis on the parametric form of the nonlinear characteristic of the Hammerstein system (Section 4),
- Using Lasso Regression for power system stability analysis, and achieving a better prediction accuracy than the existing methods in the field (Section 5).

2 Review of System Identification Methodology

2.1 System Identification and Machine Learning

The modern system identification problems arose around early 1960s in the field of control engineering, after the development of model-based control design [8], [9]. At that time, much of the control design were in actuality single-input-single-output (SISO) systems. The availability of such models finally became extended from control design in electrical and mechanical engineering [10], and they were applied in the diverse areas such as environmental systems, biological and biomedical systems, and transportation systems. Meanwhile, system identification has also been extended from being viewed as deterministic problems to being viewed as stochastic problems. [10] is one of the early works that map maximum likelihood approach from statistics into system identification, and this approach has became widely used in 1970s. Naturally, system identification later became viewed as a question of approximation theory, in the sense that one searches the model that best approximates the true physical system, rather than the model that exactly captures the true physical process. The model is then determined by the output-error model estimation method, illustrated in Fig. 1. We observe the input-output pair $(\mathbf{U}_n, Y_n)$, $1 \leq n \leq N$,

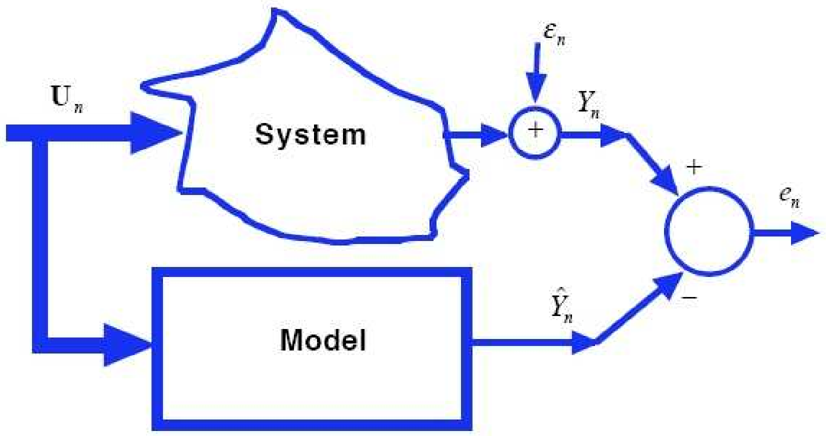


Figure 1: The output-error model-estimation method.

where input $\mathbf{U}_n \in \mathbb{R}^d$ is a stochastic process, and output $Y_n \in \mathbb{R}^1$ is mixed with some noise due to physical reasons, e.g., channel noise. Note that the input and output both can be one-dimensional or multi-dimensional, and yet we are using multiple-input-single-output (MISO) systems for illustration, for the more general case, multiple-input-multiple-output (MIMO) system can be viewed as parallel combination of several MISO systems. We suppose the same input $\{\mathbf{U}_n\}$ go through a virtual model, and the predicted output is $\{\hat{Y}_n\}$. Assume the virtual model is stationary, and has certain characteristics (e.g., linear, nonlinear, linear and nonlinear in cascade/parallel/feedback, etc) and the issue of system identification is merely to find the optimal system that minimizes a cost measure between $\{Y_n\}$ and $\{\hat{Y}_n\}$.

There are three basic approaches to system modeling:

(a) *Parametric Modeling:* $\hat{Y}_n = h(\overrightarrow{\mathbf{U}_n}|\theta)$, where $\overrightarrow{\mathbf{U}_n} = (\mathbf{U}_n, \mathbf{U}_{n-1}, \mathbf{U}_{n-2}, \cdots)$ is the history of the system input up to the time n, and $\mathbf{U}_n \in \mathbb{R}^d$.

The input-output mapping $h(\cdot)$ is specified up to the unknown, finite dimensional parameter $\theta \in \Theta$, where Θ is a set of admissible values of the parameter. That is, we assume that the model belongs to certain class of parametrically defined functions [11].

(b) *Nonparametric Modeling:* $\hat{Y}_n = h(\overrightarrow{\mathbf{U}_n})$.

The input-output mapping $h(\cdot)$ is totally unknown. That is, no assumption is imposed on the class of the system [12].

(c) *Semiparametric Modeling:* $\hat{Y}_n = h\Big(\overrightarrow{\mathbf{U}_n}|g(\cdot), \theta\Big)$.

The input-output mapping $h(\cdot)$ is specified up to the unknown, finite dimensional parameter θ and a finite set of one-dimensional functions $g(\cdot) = \Big(g_1(\cdot), \cdots g_r(\cdot)\Big)$ [12].

2.1.1 The Classical Parametric Modeling Based on the Output-Error Approach

In the parametric modeling, the output error of e_n, see Fig. 1, is given by

$$e_n(\theta) = Y_n - \hat{Y}_n(\theta) = Y_n - h(\overrightarrow{\mathbf{U}_n}|\theta).$$

We measure the expected loss by the mean square error:

$$Q(\theta) = \mathbb{E}\Big\{\Big|Y_n - h(\overrightarrow{\mathbf{U}_n}|\theta)\Big|^2\Big\}. \tag{1}$$

So for parametric modeling, the identification problem can be formulated as follows: given a training set $\{(\mathbf{U}_1, Y_1), (\mathbf{U}_2, Y_2), \cdots, (\mathbf{U}_N, Y_N)\}$ of input-output data of the unknown system, construct an algorithm $\hat{\theta}_N$ for estimating $\theta^* \in \Theta$, such that $\lim\limits_{N\to\infty} Q(\hat{\theta}_N) = Q(\theta^*)$, where $\theta^* = \arg\min\limits_{\theta\in\Theta} Q(\theta)$.

Assume without loss of generality that the input-output relationship for the true system is described by $Y_n = f^*(\overrightarrow{\mathbf{U}_n}) + \varepsilon_n$ for some unknown $f^*(\cdot)$. Also, we assume that $\mathbb{E}\{f^{*2}(\overrightarrow{\mathbf{U}_n})\} < \infty$ and $\mathbb{E}|\varepsilon_n|^2 < \infty$. Then if the noise is independent of the system history, we have:

$$\begin{aligned} Q(\theta) &= \mathbb{E}\Big\{\Big|Y_n - h(\overrightarrow{\mathbf{U}_n}|\theta)\Big|^2\Big\} \\ &= \mathbb{E}\Big\{\Big|f^*(\overrightarrow{\mathbf{U}_n}) - h(\overrightarrow{\mathbf{U}_n}|\theta)\Big|^2\Big\} + \mathbb{E}|\varepsilon_n|^2. \end{aligned} \tag{2}$$

Let us discuss two important cases of parametric modeling:

Case 1. If $f^*(\overrightarrow{\mathbf{U}_n}) = h(\overrightarrow{\mathbf{U}_n}|\theta^*)$, i.e., the true system is in the parametric model space $\mathbf{H} = \{h(\cdot|\theta) : \theta \in \Theta\}$, and the value $\theta^* \in \Theta$ defines the true system. Then under some identifiability conditions, we can expect that:

$$\theta^* = \arg\inf_{\theta\in\Theta} Q(\theta),$$

and consequently,

$$Q^*_{parametric} = \inf_{\theta\in\Theta} Q(\theta) = \mathbb{E}|\varepsilon_n|^2.$$

Thus, the noise level (measured by its variance) gives the smallest possible identification error (Bayes error).

Case 2. Let $f^*(\cdot) \notin \mathbf{H}$, i.e., the true system is outside the assumed parametric class. Then $\theta^* = \arg\inf_{\theta\in\Theta} Q(\theta)$ defines the L_2 projection of the true system onto the model class $\mathbf{H}$. Hence $h(\cdot|\theta^*)$ characterizes the best model for the given system. Note that θ^* may not be unique.

In this case, the smallest possible error for parametric modeling is given by:

$$Q^*_{parametric} = \inf_{\theta\in\Theta} Q(\theta) = \mathbb{E}\Big\{\Big|f^*(\overrightarrow{\mathbf{U}_n}) - h(\overrightarrow{\mathbf{U}_n}|\theta^*)\Big|^2\Big\} + \mathbb{E}|\varepsilon_n|^2,$$

where the first term on the right-hand-side of this expression represents the irreducible modeling error for the best possible parametric model.

In order to estimate the optimal model from given training data, we need to construct the empirical counterpart of $Q(\theta)$. A natural empirical version of $Q(\theta)$ is

$$\hat{Q}_N(\theta) = \frac{1}{N}\sum_{n=1}^{N}\Big|Y_n - h(\overrightarrow{\mathbf{U}_n}|\theta)\Big|^2.$$

As a result an estimate of θ^* can be obtained by

$$\hat{\theta}_N = \arg\min_{\theta\in\Theta} \hat{Q}_N(\theta).$$

The convergence of such estimators have been studied in some particular cases, e.g., linear situations [13]. In fact, it is known [12] that we need first to establish the following uniform convergence result,

$$P(\sup_{\theta\in\Theta}|\hat{\theta}_N(\theta) - Q(\theta)| > \delta) \to 0 \quad \text{as } N \to \infty.$$

This implies [12] that, $\forall \delta > 0$,

$$P\Big\{\Big|Q(\hat{\theta}_N) - \inf_{\theta\in\Theta} Q(\theta)\Big| > \delta\Big\} \to 0 \quad \text{as } N \to \infty,$$

and furthermore,

$$\hat{\theta}_N \to \theta^*, \quad (\mathbf{P}) \quad \text{as } N \to \infty,$$

where $(\mathbf{P})$ denotes it is convergence in probability, and also

$$h(\cdot|\hat{\theta}_N) \to h(\cdot|\theta^*), \quad (\mathbf{P}) \quad \text{as } N \to \infty.$$

Further results from machine learning theory may allow us to get a finite sample bounds for $P(\sup_{\theta\in\Theta} |\hat{Q}_N(\theta) - Q(\theta)| > \delta)$. This issue, however, will not be pursued in this thesis.

Possible distribution of errors occurring in the parametric inference of system modeling is illustrated in Fig. 2.

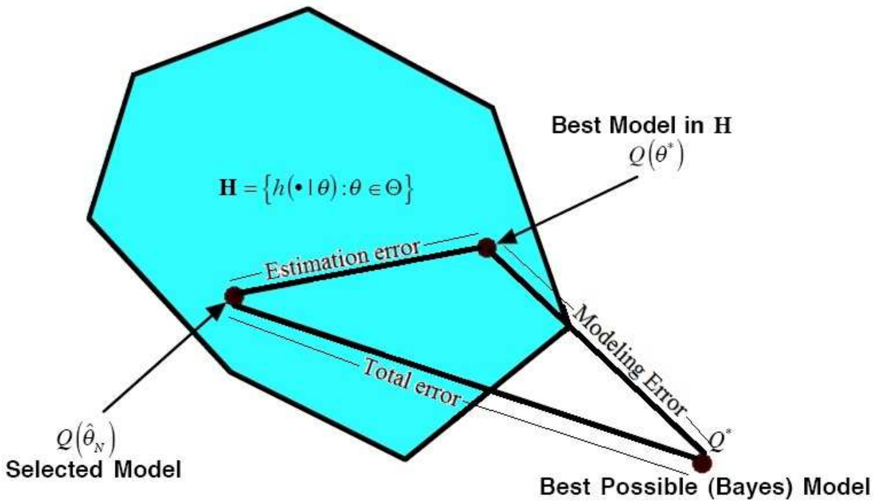


Figure 2: Relationship between errors in parametric system modeling.

2.1.2 The Nonparametric Modeling Based on the Output-Error Approach

In the fully nonparametric approach, the modeling error is eliminated completely. In fact, the minimum $h^*(\overrightarrow{\mathbf{U}_n})$ of the error

$$Q(h) = \mathbb{E}\Big\{\Big|Y_n - h(\overrightarrow{\mathbf{U}_n})\Big|^2\Big\},$$

is achieved by the regression function, i.e.,

$$h^*(\overrightarrow{\mathbf{U}_n}) = \mathbb{E}\{Y_n|\overrightarrow{\mathbf{U}_n}\}. \tag{3}$$

The corresponding minimal (Bayes) error is

$$Q(h^*) = \mathbb{E}\{\text{var}(Y_n|\overrightarrow{\mathbf{U}_n})\}.$$

For $Y_n = f^*(\overrightarrow{\mathbf{U}_n}) + \varepsilon_n$, $Q(h^*)$ is denoted as $Q^*_{nonparametric}$ is reduced to the noise level, i.e.,

$$Q^*_{nonparametric} = \mathbb{E}|\varepsilon_n|^2.$$

The distribution of errors occurring in the nonparametric inference for system modeling is shown in Fig. 3. Note that there is no modeling error.

The empirical risk, corresponding to $Q(h)$, is given by minimizing

$$\hat{Q}_N(h) = \frac{1}{N}\sum_{n=1}^{N}|Y_n - h(\overrightarrow{\mathbf{U}_n})|^2.$$

Minimization of $\hat{Q}_N(h)$, however, yields an estimate that interpolates data. A meaningful result can be obtained directly. We can, however, estimate the regression function $h^*(\overrightarrow{\mathbf{U}_n})$ assuming the system memory is finite, i.e., $Y_n = f^*(\mathbf{U}_n, \mathbf{U}_{n-1}, \cdots, \mathbf{U}_{n-p}) + \varepsilon_n$, where p is the memory size and $\mathbf{U}_n \in \mathbb{R}^d$. Note that $f^*(\cdot)$ is a $(p+1)d$-dimensional function. Then under some general conditions on the smoothness of $f^*(\cdot)$, we can construct a consistent

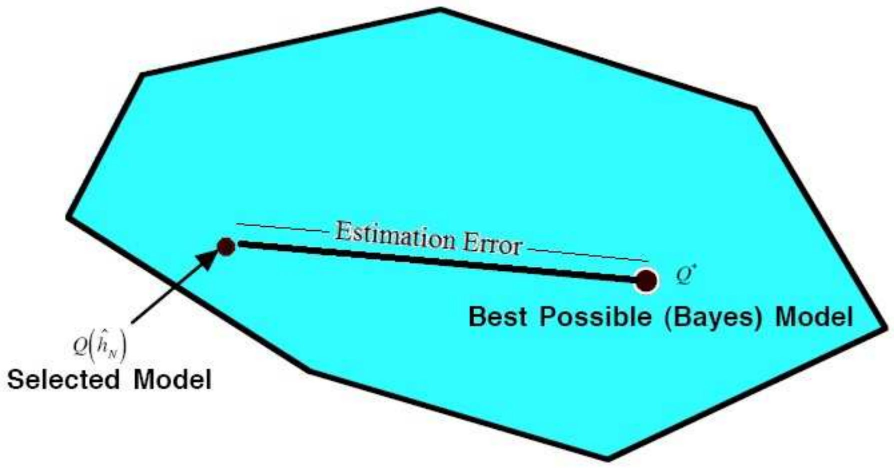


Figure 3: Relationship between errors in nonparametric system modeling.

estimate $\hat{h}_N(\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_p)$ of $f^*(\cdot)$ with the corresponding convergence rate:

$$\hat{h}_N(\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_p) = f^*(\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_p) + O_P\Big(N^{-\frac{2}{4+d(p+1)}}\Big),$$

where $O_P(\cdot)$ denotes "in probability" convergence. Clearly, this is a very slow rate of convergence for large values of d and p.

2.1.3 The Semiparametric Modeling Based on the Output-Error Approach

Parametric approach, examined in Section 2.1.1, has its inherent limitation in carrying a substantial modeling error. On the other hand, nonparametric approach, shown in Section 2.1.2, reveals a slow convergence rate for large input dimension d and large system memory p. The semiparametric strategy gives a logical tradeoff between the limitation of the parametric approach and the flexibility of nonparametric modeling.

The output-error solution in the semiparametric setting is the minimization of the criterion function

$$Q\Big(g(\cdot), \theta\Big) = \mathbb{E}\Big\{\Big|Y_n - h\Big(\overrightarrow{\mathbf{U}_n}|g(\cdot), \theta\Big)\Big|^2\Big\}$$

with respect to $g(\cdot)$ and θ.

Note that if we minimize $Q\Big(g(\cdot), \theta\Big)$ with respect to $g(\cdot)$, the solution of such problem is a set of functions $g(\cdot;\theta)$ indexed by $\theta \in \Theta$. Note that $g(\cdot;\theta)$ is the regression function

$$g(\overrightarrow{\mathbf{U}_n};\theta) = \mathbb{E}\Big\{Y_n \Big| h\Big(\overrightarrow{\mathbf{U}_n}|g(\cdot),\theta\Big)\Big\}. \tag{4}$$

Generally obtaining $g(\cdot;\theta)$ can be a difficult task. In some particular cases, examined in this thesis, the functions $g(\cdot;\theta)$ can be easily characterized. Plugging $g(\cdot;\theta)$ into $Q\Big(g(\cdot),\theta\Big)$ yields the so called *profiled risk* function which solely depends on θ

$$Q(\theta) = \mathbb{E}\Big\{\Big|Y_n - h\Big(\overrightarrow{\mathbf{U}_n}|g(\cdot;\theta),\theta\Big)\Big|^2\Big\}. \tag{5}$$

The minimum θ^* of $Q(\theta)$ characterizes the best value of the parametric model. The best nonparametric part is obtained by

$$g^*(\cdot) = g(\cdot;\theta^*).$$

As a result the optimal (in the L_2 sense) semiparametric model is given by $\Big(\theta^*, g^*(\cdot)\Big)$. The smallest possible error for semiparametric modeling is:

$$Q^*_{semiparametric} = Q\Big(g^*(\cdot),\theta^*\Big) = \mathbb{E}\Big\{\Big|f^*(\overrightarrow{\mathbf{U}_n}) - h\Big(\overrightarrow{\mathbf{U}_n}|g^*(\cdot),\theta^*\Big)\Big|^2\Big\} + \mathbb{E}|\varepsilon_n|^2.$$

Let us consider, similarly to the parametric case, the following two important cases of semiparametric modeling.

Case 1. Let $f^*(\overrightarrow{\mathbf{U}_n}) = h\Big(\overrightarrow{\mathbf{U}_n}|g^*(\cdot),\theta^*\Big)$, i.e., the true system is in the semiparametric model space

$$\mathbf{S} = \Big\{\Big(\theta, g(\cdot)\Big) : \theta \in \Theta, g(\cdot) \in \mathbf{G}\Big\},$$

where $\mathbf{G}$ is some assumed function space. The pair $\Big(\theta^*, g^*(\cdot)\Big) \in \mathbf{S}$ defines the true system. Then under some identifiability conditions we expect that $\Big(\theta^*, g^*(\cdot)\Big) =$

$\arg \inf_{(\theta, g(\cdot))\in \mathbf{S}} Q\Big(\theta, g(\cdot)\Big)$. Clearly in this case

$$Q^*_{semiparametric} = \inf_{(\theta, g(\cdot))\in \mathbf{S}} Q\Big(\theta, g(\cdot)\Big) = \mathbb{E}|\varepsilon|^2.$$

Case 2. Let $f^*(\cdot) \notin \mathbf{S}$, i.e., the true system is outside the assumed semiparametric class. Then, $\Big(\theta^*, g^*(\cdot)\Big) = \arg \inf_{(\theta, g(\cdot))\in \mathbf{S}} Q\Big(\theta, g(\cdot)\Big)$ defines the L_2 projection of the system onto the model class $\mathbf{S}$. Hence $\Big(\theta^*, g^*(\cdot)\Big)$ characterize the best model $h\Big(\cdot\,|g^*(\cdot), \theta^*\Big)$, for the given true system. The smallest possible error for semiparametric modeling is given by

$$Q^*_{semiparametric} = Q\Big(g^*(\cdot), \theta^*\Big) = \mathbb{E}\Big\{\Big|f^*(\overrightarrow{\mathbf{U}_n}) - h\Big(\overrightarrow{\mathbf{U}_n}|g^*(\cdot), \theta^*\Big)\Big|^2\Big\} + \mathbb{E}|\varepsilon_n|^2,$$

where the first term represents the irreducible modeling error for the best possible semiparametric model.

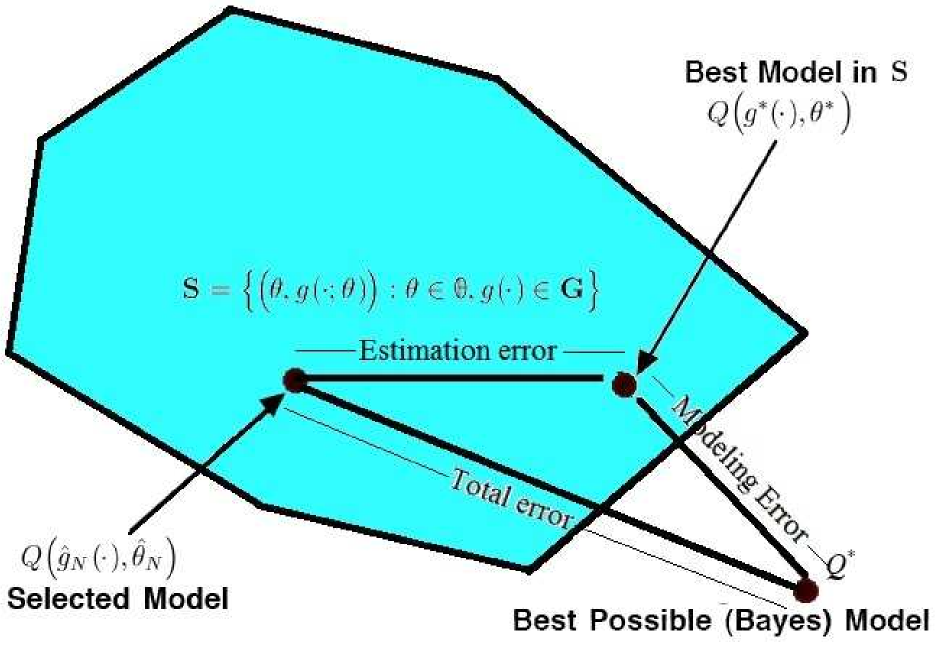

Best Model in S
$Q\Big(g^*(\cdot), \theta^*\Big)$
$\mathbf{S} = \Big\{\Big(\theta, g(\cdot;\theta)\Big) : \theta \in \Theta, g(\cdot) \in \mathbf{G}\Big\}$
Estimation error
Modeling Error
Total error
Q^*
$Q\Big(\hat{g}_N(\cdot), \hat{\theta}_N\Big)$
Selected Model
Best Possible (Bayes) Model

Figure 4: Relationship between errors in semiparametric system modeling.

The possible distribution of errors occurring in semiparametric estimation is illustrated in Fig. 4.

There is a fundamental relationship between the aforementioned three modeling strategies: parametric, nonparametric, and semiparametric,

$$Q^*_{nonparametric} \leq Q^*_{semiparametric} \leq Q^*_{parametric}. \tag{6}$$

In order to estimate the empirical characteristic of the semiparametric model, we need to estimate $g(\cdot;\theta)$ in (4). Next we use this estimate in the empirical version of the profile error $Q(\theta)$ defined in (5). Hence, denoting a generic estimate of $g(\cdot;\theta)$ by $\hat{g}_N(\cdot;\theta)$, we can use the following empirical version of $Q(\theta)$.

1. Resubstitution estimate: $\hat{Q}_N(\theta) = \frac{1}{N}\sum_{n=1}^{N}\left|Y_n - h\left(\overrightarrow{\mathbf{U}_n}|\hat{g}_N(\cdot;\theta),\theta\right)\right|^2$. This is the resubstitution error that typically is not a very accurate estimate of $Q(\theta)$ as it employs the same data for estimating $g(\cdot;\theta)$ and the evaluation of the error.

2. Partition estimate: $\tilde{Q}_N(\theta) = \frac{1}{N_2}\sum_{T_2}\left|Y_n - h\left(\overrightarrow{\mathbf{U}_n}|\hat{g}_{N_1}(\cdot;\theta),\theta\right)\right|^2$. Here $\hat{g}_{N_1}(\cdot;\theta)$ is the estimate of $g(\cdot;\theta)$ obtained from the subset T_1 of the training set size of N_1. The remaining part of data T_2, size of N_2, is employed for the evaluation of $Q(\theta)$. This partition estimate is usually very precise, because it facilitates not only the mathematical analysis of the estimation algorithms but also gives a desirable separation of parametric and nonparametric estimation problems, which allows one to evaluate parametric and nonparametric estimates more efficiently [12].

3. Cross-Validation methods, e.g., leave-one-out method: $\hat{Q}_N(\theta) = \frac{1}{N}\sum_{n=1}^{N}\Big|Y_n - h\left(\overrightarrow{\mathbf{U}_n}|\hat{g}_{-n}(\cdot;\theta),\theta\right)\Big|^2$, where $\hat{g}_{-n}(\cdot;\theta)$ is the estimate of $g(\cdot;\theta)$ using the data set where $(\mathbf{U}_n, Y_n)$ is deleted.

It is also worth noting that although the asymptotic Bayes estimation error for parametric, nonparametric, and semiparametric approaches have the relationship in (6), it is hard to compare $Q(\hat{\theta}_N)$, $Q(\hat{h}_N)$ and $Q\left(\hat{g}_N(\cdot),\hat{\theta}_N\right)$, the estimators based on finite data cases.

2.2 Block-Oriented Systems

In signal processing, linear time-invariant (LTI) systems are often used to model the characteristics of filters and simple circuits. However, in many other cases, linearity assumption may not be sufficient, and nonlinearity needs to be taken into consideration in order to capture more general cases of applications. The so-called block-oriented system is a general class of systems which combines linear dynamic subsystems as well as static nonlinear subsystems. The combination of these subsystems could be either in cascade, or in parallel, or both. Fig. 5 shows some examples of popular forms of block-oriented systems.

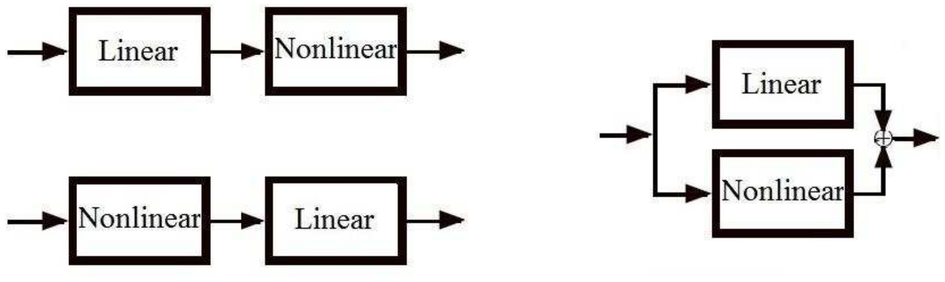


Figure 5: Examples of block-oriented system.

Block-oriented systems are a powerful and parsimonious trade-off between linear models and general nonlinear models, and are important in many cases [14]. Quite often, they are able to reflect the nature of many physical applications, e.g., biological applications, neological process, brain theory, etc [15].

System modeling can be studied in either discrete-time or continuous-time perspective. Although analog system modeling is closer to the nature of physical process, for the reason of complexity, most of research in this area are focused on discrete-time modeling.

Out of the many possible block-oriented systems, Hammerstein system is a popular one [16], [17], [18], [19]. It is named after the German mathematician Hammerstein, who examined nonlinear integral equations called “Hemmerstein integral equations” [20]. The model consists of a nonlinear subsystem, followed by a linear one. Fig. 6 is an example showing a single-input-single-output (SISO) Hammerstein system with a general nonlinear subsystem $\hat{m}^*(\cdot)$ and an infinite impulse response (IIR) linear subsystem, represented by $\{\lambda_i^*\}$.

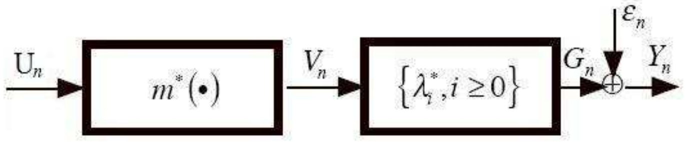


Figure 6: SISO Hammerstein model with general univariate nonlinearity and infinite impulse response dynamical subsystem.

In the above figure, ε_n is the nonlinear noise inside the model. Fig. 6 corresponds to the following relationship between the input, output and intermediate signals, where "$*$" denotes the true characteristics:

$$V_n = m^*(U_n), \quad G_n = \sum_{i=0}^{\infty} \lambda_i^* V_{n-i}, \quad Y_n = G_n + \varepsilon_n.$$

Similar to Hammerstein system, Wiener system is another commonly used block-oriented system, it composed a linear subsystem and a nonlinear subsystem afterwards. Fig. 7 depicts Wiener system.

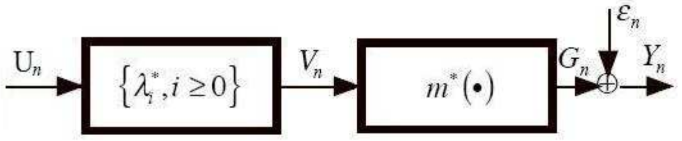


Figure 7: SISO Wiener model with general univariate nonlinearity and infinite impulse response dynamical subsystem.

Another block-oriented system which has been extensively studied is the so-called Sandwich model, which is a combination of Hammerstein and Wiener models. Fig. 8 shows a SISO Sandwich model.

This thesis will mainly focus on identifying Hammerstein systems. However, it can be conjectured that similar techniques can be applied to Wiener systems with modifications.

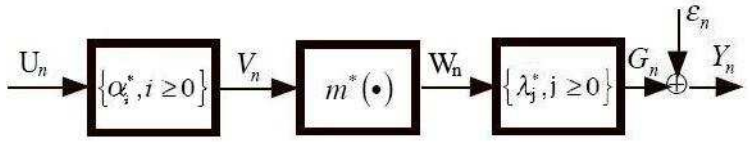


Figure 8: SISO Sandwich model.

Based on the dimensionality of input/output, block-oriented systems can be categorized into single-input-single-output (SISO), multiple-input-single-output (MISO) and multiple-input-multiple-output (MIMO) systems. Since MIMO systems can be viewed as parallel combination of MISO systems, MISO systems are often studied in the existing research.

2.3 Identification of MISO Hammerstein Models

The MISO Hammerstein system is shown in Fig. 9. The input, output relationship is given

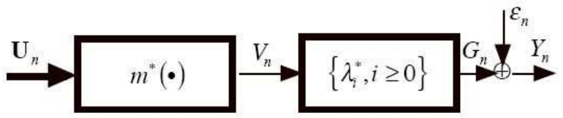


Figure 9: MISO Hammerstein model with general multivariate nonlinearity and infinite impulse response dynamical subsystem

by Fig. 9.

$$V_n = m^*(\mathbf{U}_n), \quad G_n = \sum_{i=0}^{\infty} \lambda_i^* V_{n-i}, \quad Y_n = G_n + \varepsilon_n,$$

where $\mathbf{U}_n = (U_{n,1}, U_{n,2}, \cdots, U_{n,d})^T \in \mathbb{R}^d$ is the d-dimensional input to the system. The intermediate signals V_n and G_n are not observed.

2.3.1 Identification of Linear Subsystem

In identifying a Hammerstein system, we normalize the linear subsystem such that $\lambda_0^* = 1$. Furthermore, we assume $\{\mathbf{U}_n\}$ to be a sequence of independent identically distributed vectors.

Let us first consider the single-input Hammerstein system. It is worth mentioning that the linear subsystem inside the Hammerstein model is much simpler to estimate compared with identifying the nonlinear part. For instance, correlation method provides a simple technique to estimate the linear part. In fact, the following relationship holds, see Section 2.4 for the proof.

$$\lambda_i^* = \frac{\text{cov}(Y_{n+i}, U_n)}{\text{cov}(Y_n, U_n)}. \tag{7}$$

In the case of multiple-input Hammerstein system , the formula in (7) must be modified. Hence, we have

$$\lambda_i^* = \frac{\text{cov}(Y_{n+i}, \eta(\mathbf{U}_n))}{\text{cov}(Y_n, \eta(\mathbf{U}_n))}, \tag{8}$$

where $\eta : \mathbb{R}^d \to \mathbb{R}$ is any function selected by the user, such that $\mathbb{E}(m^*(\mathbf{U}_n))\eta(\mathbf{U}_n)) \neq 0$. For instance $\eta(\mathbf{U}) = \frac{\sum_{i=1}^d \eta_i(U_i)}{d}$, for some function $\eta_i(\cdot)$, $i = 1, \cdots, d$.

The correlation formulas (7), (8) suggest that λ_i^* can be estimated by:

$$\hat{\lambda}_i = \frac{\frac{1}{N}\sum_{j=1}^{N-i}(Y_{j+i}U_j)}{\frac{1}{N}\sum_{j=1}^{N}(Y_j U_j)}, \tag{9}$$

for the one-dimensional input, where N is the size of training set, and

$$\hat{\lambda}_i = \frac{\frac{1}{N}\sum_{j=1}^{N-i}(Y_{j+i}\eta(\mathbf{U}_j))}{\frac{1}{N}\sum_{j=1}^{N}(Y_j \eta(\mathbf{U}_j))}, \tag{10}$$

for the multi-dimensional case, it has been demonstrated [12] that:

$$\hat{\lambda}_i = \lambda_i^* + O_P(N^{-1/2}), \tag{11}$$

Since identifying the linear part is numerically simple. In our research, we focus mostly on the problem of recovering the nonlinear subsystem. Hence, we may assume that $\hat{\lambda}_i$ is close to its true value λ_i^*, and in the context of nonlinear subsystem identification, we use, without loss of generality, λ^* instead of $\hat{\lambda}$.

2.3.2 Identification of the Nonlinear Subsystem

Generally, parametric and nonparametric approaches can be applied to identifying the nonlinear part of the Hammerstein model. Parametric methods require prior information on the characteristic of the subsystems, therefore by its nature it always carries a risk of modeling error. On the other hand, the nonparametric approach requires no prior assumption on the class of functions defining the nonlinear subsystem.

Under the assumption $\mathbb{E}\Big\{m^*(\mathbf{U})\Big\} = 0$ and the normalization $\lambda_0^* = 1$, it can be shown [12] that optimized by mean square error,

$$m^*(\mathbf{u}) = \mathbb{E}(Y_n|\mathbf{U}_n = \mathbf{u}). \tag{12}$$

The proof of this identity is also presented in Section 2.4. Based on this regression relationship in (12), various nonparametric techniques can be applied for recovering $m^*(\mathbf{u})$. Some nonparametric estimates are listed below:

- Classical kernel estimate

  $$\hat{m}(\mathbf{u}) = \frac{\sum_{i=1}^N Y_i K(\frac{||\mathbf{u}-\mathbf{U}_i||}{h})}{\sum_{i=1}^N K(\frac{||\mathbf{u}-\mathbf{U}_i||}{h})}, \tag{13}$$

  where $K(\cdot)$ is a d-dimensional kernel function, e.g., Gaussian kernel, and h is a smoothing parameter (bandwidth). When input $\{U_n\}$ to the system is one-dimensional, (13) becomes:

  $$\hat{m}(u) = \frac{\sum_{i=1}^N Y_i K(\frac{u-U_i}{h})}{\sum_{i=1}^N K(\frac{u-U_i}{h})}.$$

- Local linear kernel estimate.

  For one-dimensional input, local kernel estimate has the following form:

  $$\hat{m}(u) = \frac{\sum_{i=1}^N Y_i K(\frac{u-U_i}{h})}{\sum_{i=1}^N K(\frac{u-U_i}{h})} + \Big(u - \bar{U}(u)\Big)\frac{\sum_{i=1}^N Y_i\Big(U_i - \bar{U}(u)\Big)K\Big(\frac{u-U_i}{h}\Big)}{\sum_{i=1}^N Y_i\Big(U_i - \bar{U}(u)\Big)^2 K\Big(\frac{u-U_i}{h}\Big)}, \tag{14}$$

where

$$\bar{U}(u) = \frac{\sum_{i=1}^{N} U_i K(\frac{u-U_i}{h})}{\sum_{i=1}^{N} K(\frac{u-U_i}{h})}$$

is the local weighted mean of $\{U_i\}$.

For d-dimensional input, see [21] for the explicit expression of local linear kernel estimator.

- Convolution kernel estimate.

  When input to the system is one-dimensional and bounded on the support $[a, b]$, the convolution kernel estimate can be applied in nonparametric estimation. $U_{(1)}, U_{(2),\cdots}, U_{(n)}$ is the ordered version of $U_1, U_{2,\cdots}, U_n$, $U_{(0)} = a$ and $U_{(N+1)} = b$ are defined to be boundary values of the support of the input distribution, and $\Big\{(U_{(1)}, Y_{[1]}), \cdots, (U_{(N)}, Y_{[N]})\Big\}$ is input-output pair from the training set $\Big\{(U_1, Y_1) \cdots, (U_N, Y_N)\Big\}$. Then the convolution kernel estimate has the following form:

$$\hat{m}(u) = \sum_{i=1}^{N+1} Y_{[i]}(U_{(i)} - U_{(i-1)})h^{-1}K\Big(\frac{u-U_{(i)}}{h}\Big), \tag{15}$$

  or

$$\hat{m}(u) = \sum_{i=1}^{N} Y_{[i]}\frac{(U_{(i+1)} - U_{(i-1)})}{2}h^{-1}K\Big(\frac{u-U_{(i)}}{h}\Big), \tag{16}$$

  Note that $Y_{[i]}$'s are not ordered, they are just in pair with $U_{(i)}$'s. See Section 2.4 for more details. Compared with other kernel estimates, convolution kernel has the advantage of not having denominator. However, it can only be applied to 1-dimensional input cases, since there is no multivariate counterpart for ordering the data.

- Orthogonal series estimate

$$\hat{m}(\mathbf{u}) = \frac{\sum_{k=0}^{T} \hat{a}_k \psi_k(\mathbf{u})}{\sum_{k=0}^{T} \hat{b}_k \psi_k(\mathbf{u})}, \tag{17}$$

  where $\Big\{\psi_i(\cdot)\Big\}$ is an orthonormal basis system defined on the support D that includes the support of $\mathbf{U}_n$, i.e.:

$$\int_D \psi_i(\mathbf{u})\psi_j(\mathbf{u})d\mathbf{u} = \begin{cases} 1, & \text{for } i = j \\ 0, & \text{for } i \neq j, \end{cases}$$

and T is the truncation parameter, and

$$\hat{a}_k = \frac{1}{N}\sum_{n=1}^{N} Y_n \psi(\mathbf{U}_n),$$

$$\hat{b}_k = \frac{1}{N}\sum_{n=1}^{N} \psi(\mathbf{U}_n).$$

Note that the denominator in (17) estimates the input density $f_{\mathbf{U}}(\mathbf{u})$, and the numerator estimates the term $m^*(\mathbf{u})f_{\mathbf{U}}(\mathbf{u})$.

For identifying the nonlinear subsystem inside a MISO Hammerstein model, it can be demonstrated [12] that by using nonparametric estimate,

$$\hat{m}(\mathbf{u}) = m^*(\mathbf{u}) + O_P(N^{-\frac{s}{2s+d}}), \tag{18}$$

where d is the dimension of input data, and s indicates the smoothness of the function $m^*(\cdot)$ measured in number of existing derivatives. The formula in (18) shows that the convergence rate of nonparametric estimate decreases as the dimension of input increases. For example, suppose $m^*(\cdot)$ has finite second derivative, i.e., $s = 2$, and we examine $|\hat{m}(\mathbf{u}) - m^*(\mathbf{u})|$ for different value of d. Suppose length of data N_1 is needed for $d = 1$ and $|\hat{m}(u) - m^*(u)| < \delta$. On the other hand, length of data N_d is needed for a d-dimensional system to achieve the same level of estimation error δ, then we have

$$N_d^{-\frac{2}{d+4}} \simeq N_1^{-\frac{2}{5}}, \tag{19}$$

that is,

$$N_d \simeq N_1^{\alpha \cdot d},$$

where $\alpha = \frac{d+4}{5d}$. For different value of d, $\frac{1}{5} \leq \alpha \leq 1$. It shows that to obtain a given degree of precision of a nonparametric estimate, the sample size must grow exponentially with the input dimension d. This illustrates the so-called “Curse of dimensionality” in nonparametric estimation.

2.4 Auxiliary Proofs

The following proofs can be also found in [12].

Proofs of (7) and (8) When the input $\{U_n\}$ is one dimensional, according to Fig. 6, the following relationship between input and output signal holds:

$$Y_n = \sum_{j=0}^{\infty} \lambda_j^* m^*(U_{n-j}) + \varepsilon_n,$$

therefore:

$$\begin{aligned} \text{cov}(Y_{n+i}, U_n) &= \text{cov}\Big(\sum_{j=0}^{\infty} \lambda_j^* m^*(U_{n+i-j}) + \varepsilon_{n+i},\ U_n\Big) \\ &= \text{cov}(\lambda_i^* m^*(U_n), U_n) + \text{cov}(\sum_{j=0, j\neq i}^{\infty} \lambda_j^* m^*(U_{n+i-j}), U_n) \\ &= \lambda_i^* \text{cov}(m^*(U_n), U_n) + \sum_{j=0, j\neq i}^{\infty} \lambda_j^* \text{cov}(m^*(U_{n+i-j}), U_n). \end{aligned}$$

Since $\{U_n\}$ is an i.i.d process, $\text{cov}(m^*(U_{n+i-j}), U_n) = 0$ for $i \neq j$. Then we have $\text{cov}(Y_{n+i}, U_n) = \lambda_i^* \text{cov}(m^*(U_n), U_n)$. Since $\lambda_0^* = 1$ and $\text{cov}(Y_n, U_n) = \lambda_0^* \text{cov}(m^*(U_n), U_n)$, we get $\lambda_i^* = \frac{\text{cov}(Y_{n+i}, U_n)}{\text{cov}(Y_n, U_n)}$. Thus the relationship (7) has been proved.

When input $\{\mathbf{U}_n\}$ is multi-dimensional, Fig. 9 yields the following relationship between input and output:

$$Y_n = \sum_{j=0}^{\infty} \lambda_j^* m^*(\mathbf{U}_{n-j}) + \varepsilon_n.$$

From this we get

$$\begin{aligned}
\text{cov}(Y_{n+i}, \eta(\mathbf{U}_n)) &= \text{cov}\Big(\sum_{j=0}^{\infty} \lambda_j^* m^*(\mathbf{U}_{n+i-j}) + \varepsilon_{n+i},\ \eta(\mathbf{U}_n)\Big) \\
&= \text{cov}(\lambda_i^* m^*(\mathbf{U}_n), \eta(\mathbf{U}_n)) + \text{cov}(\sum_{j=0, j\neq i}^{\infty} \lambda_j^* m^*(\mathbf{U}_{n+i-j}), \eta(\mathbf{U}_n)) \\
&= \lambda_i^* \text{cov}(m^*(\mathbf{U}_n), \eta(\mathbf{U}_n)) + \sum_{j=0, j\neq i}^{\infty} \lambda_j^* \text{cov}(m^*(\mathbf{U}_{n+i-j}), \eta(\mathbf{U}_n)).
\end{aligned}$$

Since $\{\mathbf{U}_n\}$ is an i.i.d process, $\text{cov}(m^*(\mathbf{U}_{n+i-j}), \eta(\mathbf{U}_n)) = 0$ for $i \neq j$. Then we have $\text{cov}(Y_{n+i}, \eta(\mathbf{U}_n)) = \lambda_i^* \text{cov}(m^*(\mathbf{U}_n), \eta(\mathbf{U}_n))$. Because $\lambda_0^* = 1$, and $\text{cov}(Y_n, \eta(\mathbf{U}_n)) = \lambda_0^* \text{cov}(m^*(\mathbf{U}_n), \eta(\mathbf{U}_n))$, then $\lambda_i^* = \frac{\text{cov}(Y_{n+i}, \eta(\mathbf{U}_n))}{\text{cov}(Y_n, \eta(\mathbf{U}_n))}$. Thus the relationship (8) has been proved.

Proof of (12) Since $\{\mathbf{U}_n\}$ is a white process, relation $\mathbb{E}(m^*(\mathbf{U}_i)|\mathbf{U}_0) = \mathbb{E}(m^*(\mathbf{U}_i))$ holds for $i \neq 0$. Thus,

$$\begin{aligned}
\mathbb{E}(Y_n|\mathbf{U}_n = \mathbf{u}) &= \mathbb{E}(\sum_{j=0}^{\infty} \lambda_j^* m^*(\mathbf{U}_{n-j}) + \varepsilon_n|\mathbf{U}_n = \mathbf{u}) \\
&= \lambda_0^* m^*(\mathbf{u}) + \mathbb{E}m^*(\mathbf{U}) \sum_{i=1}^{\infty} \lambda_i^*.
\end{aligned}$$

Under the condition $\mathbb{E}m^*(\mathbf{U}) = 0$ and $\lambda_0^* = 1$, we have $\mathbb{E}(Y_n|\mathbf{U}_n = \mathbf{u}) = m^*(\mathbf{u})$, therefore (12) has been proved.

The aforementioned considerations imply that

$$\begin{aligned}
Y_n &= m^*(\mathbf{U}_n) + \sum_{i=1}^{\infty} \lambda_i^* m^*(\mathbf{U}_{n-i}) + \varepsilon_n \\
&= m^*(\mathbf{U}_n) + \xi_n + \varepsilon_n, \qquad (20)
\end{aligned}$$

where $\xi_n = \sum_{i=1}^{\infty} \lambda_i^* m^*(\mathbf{U}_{n-i})$ is independent of $\mathbf{U}_n$, and can be viewed as "system noise" due to the memory of system, in contrast with ε_n, the external measurement noise.

Proofs of (15) and (16) When $h \to 0$, the convolution property of smoothing kernels (See [12] for admissible class of such kernels) gives that

$$\frac{1}{h}\int_a^b m^*(z)K(\frac{u-z}{h})dz \to m^*(u),$$

at every points of u where $m^*(u)$ is continuous. Here we assume that the support of $m^*(u)$ is the interval $[a, b]$. Since according to (20), $Y_n = m^*(U_n) + \xi_n$, then ordering the input of data such that $a = U_{(0)} \leq U_{(1)} \leq \cdots, \leq U_{(N)} \leq U_{(N+1)} = b$, we obtain

$$\begin{aligned}
\frac{1}{h}\int_a^b m^*(z)K(\frac{u-z}{h})dz &= \frac{1}{h}\int_{U_{(0)}}^{U_{(1)}} m^*(z)K(\frac{u-z}{h})dz + \frac{1}{h}\int_{U_{(1)}}^{U_{(2)}} m^*(z)K(\frac{u-z}{h})dz + \cdots \\
&\quad + \frac{1}{h}\int_{U_{(n)}}^{U_{(n-1)}} m^*(z)K(\frac{u-z}{h})dz + \cdots \frac{1}{h}\int_{U_{(N)}}^{U_{(N+1)}} m^*(z)K(\frac{u-z}{h})dz \\
&\simeq \sum_{j=1}^{N} m^*(U_{(j)})\frac{1}{h}\int_{U_{(j-1)}}^{U_{(j)}} K(\frac{u-z}{h})dz \\
&\simeq \sum_{j=1}^{N} Y_{[j]}\frac{1}{h}\int_{U_{(j-1)}}^{U_{(j)}} K(\frac{u-z}{h})dz.
\end{aligned}$$

Notice that $\int_{U_{(j-1)}}^{U_{(j)}} K(\frac{u-z}{h})dz$ can be either approximated by $(U_{(j)} - U_{(j-1)})K(\frac{u-U_{(j)}}{h})$ or by $\frac{(U_{(j+1)}-U_{(j-1)})}{2}K(\frac{u-U_{(j)}}{h})$. Therefore the relationships in (15) and (16) have been proved.

3 Semiparametric Hammerstein Model Identification

3.1 Semiparametric MISO Hammerstein Model

In identifying a MISO Hammerstein Model (Fig. 9 in Section 2.3), nonparametric estimate of the system with larger dimensions of input, usually has a slower convergence rate, see (18). Particularly, for very large d, the estimate $\hat{m}(\cdot)$ hardly converges. This is referred to as “Curse of dimensionality”.

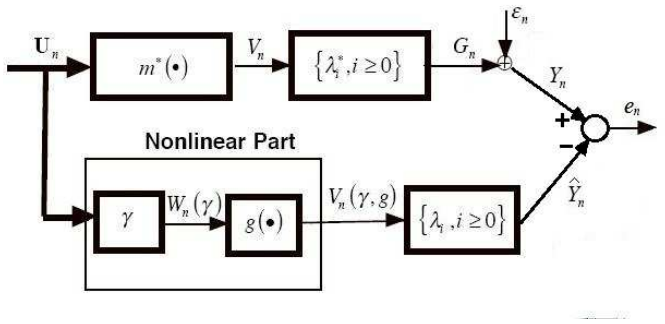


Figure 10: Semiparametric approximation of a MISO Hammerstein system.

In order to overcome this disadvantage in nonparametric modeling, semiparametric Hammerstein models have been proposed [12], [22] to approximate MISO Hammerstein system, see Fig. 10. The intention is to replace the multivariate input function $m^*(\cdot)$ by the approximation $m^*(\mathbf{u}) \approx g^*(\gamma^{*T}\mathbf{u})$. Note we need to assume $\mathbb{E}(\mathbf{U}) = 0$, and $\lambda_0^* = 1$ as before, and also $\sum_{j=0}^{\infty} |\lambda_j^*| < \infty$. Then the relationship between input, output and intermediate signals are:

$$W_n(\gamma) = \gamma^T \mathbf{U}_n, \;\; V_n(\gamma, g) = g(\gamma^T \mathbf{U}_n), \;\; \hat{Y}_n = \sum_{j=0}^{\infty} \lambda_j V_{n-j}(\gamma, g).$$

The output-error model-estimation method applied to the system and the model leads to the following minimization:

$$(\tilde{\lambda}^*, \gamma^*, g^*(\cdot)) = \arg \min_{(\lambda, \gamma, g(\cdot))}, Q(\lambda, \gamma, g(\cdot)) \tag{21}$$

where $\lambda = (\lambda_0, \lambda_1, \cdots)^T$, and

$$Q(\lambda, \gamma, g(\cdot)) = \mathbb{E}\Big| Y_n - \sum_{i=0}^{\infty} \lambda_i g(\gamma^T \mathbf{U}_{n-i}) \Big|^2. \tag{22}$$

if the loss is measured by mean-square criterion.

It can be proved (see Section 3.2.4) that

$$\tilde{\lambda}^* = \lambda^* = (\lambda_0^*, \lambda_1^*, \cdots)^T, \tag{23}$$

which means the value of λ that leads to the smallest modeling error is actually the true value λ^* itself. Since λ^* can be estimated by correlation method (9), our focus will be mainly on estimation of γ^* and $g^*(\cdot)$, and assuming the estimate $\hat{\lambda}$ to be close to λ^*, we use the value λ^* directly in the following studies. Note that $(\gamma^*, g^*(\cdot))$ may not be unique, but it defines the semiparametric Hammerstein model that is the closed to the true MISO Hammerstein system. In the case that $m^*(\mathbf{u}) = g^*(\gamma^{*T}\mathbf{u})$, that is, when the MISO Hammerstein system can be exactly represented by semiparametric model, the system can be shown in Fig. 10. By using semiparametric model for MISO Hammerstein systems, the

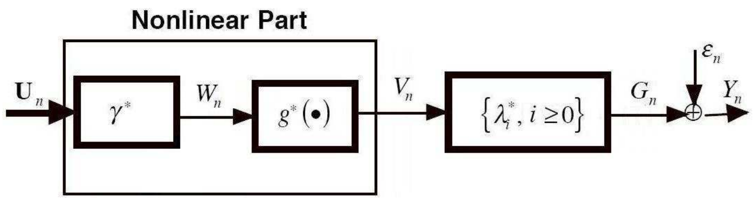


Figure 11: Semiparametric Hammerstein system.

convergence rate for estimating nonparametric part is $N^{-2/5}$, see (19), for any dimension of the input. The essence of semiparametric model is the projection of a the multivariate nonlinear function onto a one dimension function space. Semiparametric model is somewhere between parametric case where the nonlinear function is fully parameterized, and the fully nonparametric case where the class of nonlinear function is totally unspecified.

3.2 Algorithms for Semiparametric Hammerstein Model Identification

The main characteristics to estimate are the parametric characteristic γ^* and the nonparametric characteristic $g^*(\cdot)$. Furthermore, we assume the linear subsystem is a finite order impulse response (FIR) of order p. For IIR cases, we can always approximate them by a FIR subsystem where p can be viewed as a truncation parameter. In this section we assume $m^*(\mathbf{u}) = g^*(\gamma^{*T}\mathbf{u})$, the semiparametric model exactly describes the true MISO Hammerstein model, see Fig. 11). Then the input-output relationship is the following:

$$W_n = \gamma^{*T}\mathbf{U}_n, \;\; V_n = g^*(W_n), \;\; G_n = \sum_{i=0}^{\infty} \lambda_i^* V_{n-i}, \;\; Y_n = G_n + \varepsilon_n.$$

Note that the identification algorithms to identify such system can also be applied to a more general case, when the semiparametric model is only an approximation to the true MISO Hammerstein system, which will be discussed in Section 3.3.

3.2.1 Parameter Estimation

Parametric Inference Based on Minimizing the Empirical Mean Squared Error

In order to identify γ^* and $g^*(\cdot)$, we need to estimate them one at a time. i.e., assume γ is specified, then $\{W_n\}$ is fully specified. Denote this series of intermediate signal by $\{W_n(\gamma)\}$, since it is fully dependent on γ. Notice the model between $\{W_n(\gamma)\}$ and $\{Y_n\}$ is a SISO Hammerstein model, thus $g(\cdot)$ can be estimated through nonparametric techniques, denote this estimate by $\hat{g}(\cdot;\gamma)$. The above procedure can be demonstrated by Fig. 12.

Using another data set to evaluate the estimation error, it is shown that $\gamma = \gamma^*$ is the asymptotic minimizer of this error [12]. Thus based on a given data set of input observations and output responses, estimation of γ^* would require some sub-sampling schemes, e.g., leave-one-out, Cross-Validation, partition method, etc. In this research, for the most part we use partition method, that is we minimize the criterion function, the empirical pro-

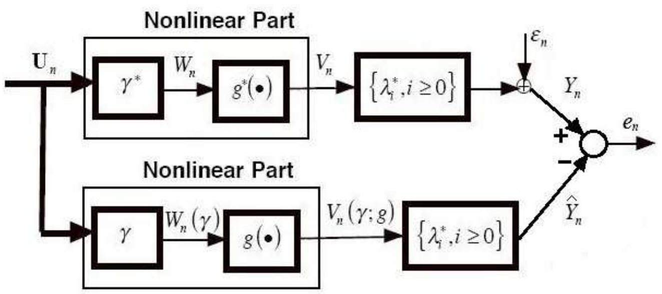


Figure 12: The block diagram illustrating the semiparametric approach.

file risk function:

$$\hat{Q}_N = \frac{1}{N_2}\sum_{T_2}(Y_n - \sum_{i=0}^{p}\hat{\lambda}_i\hat{g}(W_{n-i}(\gamma);\gamma))^2, \tag{24}$$

with respect to γ, the minimum is given by

$$\hat{\gamma} = \arg\min_{\gamma\in\Gamma}\hat{Q}_N(\gamma).$$

It is worth mentioning that (24) is the most general formula involving the estimate $\hat{\lambda}$ of λ^*. In simulation studies, however, the true value λ^* is employed since $\hat{\lambda}$ is the consistent estimate and as such does not have an essential influence on the overall accuracy of the Hammerstein system identification algorithm. Note T_2 is a subset of given data, and $\hat{g}(w;\gamma)$ is an nonparametric estimator based on T_1, another subset of given data. Note that data in T_1 and T_2 should be totally independent of each other. Obtaining $\hat{g}(w;\gamma)$ could be by any one of the nonparametric estimation formulas, e.g., in our research, we use local linear kernel estimator:

$$\hat{g}(w;\gamma) = \frac{\sum_{T_1} Y_n K\left(\frac{w-W_n(\gamma)}{h}\right)}{\sum_{T_1} K\left(\frac{w-W_n(\gamma)}{h}\right)} + \left(w - \bar{W}(w;\gamma)\right)\frac{\sum_{T_1} Y_n\left(W_n(\gamma)-\bar{W}(w;\gamma)\right)K\left(\frac{w-W_n(\gamma)}{h}\right)}{\sum_{T_1}\left(W_n(\gamma)-\bar{W}(w;\gamma)\right)^2 K\left(\frac{w-W_n(\gamma)}{h}\right)}, \tag{25}$$

and

$$\bar{W}(w;\gamma) = \frac{\sum_{T_1} W_n(\gamma)K\left(\frac{w-W_n(\gamma)}{h}\right)}{\sum_{T_1} K\left(\frac{w-W_n(\gamma)}{h}\right)}$$

is the local weighted mean of $W_n(\gamma)$.

The partition of data set is shown in Fig. 13.

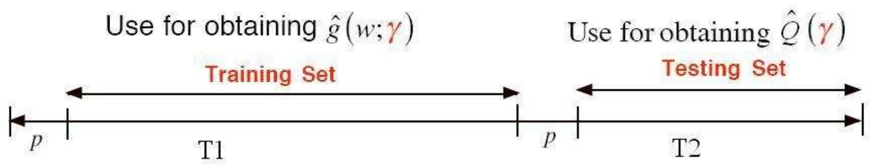

Use for obtaining $\hat{g}(w;\gamma)$
Training Set
Use for obtaining $\hat{Q}(\gamma)$
Testing Set
p
T1
p
T2

Figure 13: Partitioning data into training and testing subsets. Note that the first p observations in the data set are deleted. Also T_1 and T_2 are p observations away from each other so as to make sure (24) uses independent data for "training" and "testing".

Using the methodology developed in [12] [23], we can conjure that the estimator $\hat{\gamma}$ converges in probability to γ^* as $N_1 \to \infty$ and $N_2 \to \infty$.

Direct Parameter Estimation via the Average Derivative Technique

The aforementioned algorithm for estimating $\hat{\gamma}$ requires finding the minimum of the empirical mean squared criterion. This calls for some efficient optimization scheme. Yet such an estimate is not explicit. Another method which directly estimates γ^* is based on the following strategy. Let the conditions

$$\mathbb{E}\{g^*(W)\} = 0 \quad \text{and} \quad \mathbb{E}\{g^{*(1)}(W)\} \neq 0$$

be satisfied, then:

$$\frac{\gamma_j^*}{\gamma_1^*} = \frac{\mathbb{E}(Y_n l_j(\mathbf{U}_n))}{\mathbb{E}(Y_n l_1(\mathbf{U}_n))}, \qquad j = 2, \cdots, d, \tag{26}$$

where $\mathbf{U}_n = \Big(U_{n,1}, \cdots, U_{n,d}\Big)$ is the multivariate input, $l_j(\mathbf{u}) = \frac{\partial f(\mathbf{u})/\partial u_j}{f(\mathbf{u})}$ and $f(\mathbf{u})$ is the probability density function of $\mathbf{U}_n$. Section 3.2.4 gives briefly the details of this method.

For the input signals that have unknown density, $f(\mathbf{u})$ can be estimated through kernel methods, see [12]. Particularly if we have prior knowledge about input density, e.g., if $\mathbf{U}_n$ is $N_d(0, \Sigma)$ with $\Sigma = diag(\sigma_1^2, \cdots, \sigma_d^2)$, then the derivative method based on (26) becomes:

$$\frac{\gamma_j^*}{\gamma_1^*} = \frac{\sigma_1^2}{\sigma_j^2} \frac{\mathbb{E}\Big\{Y_n U_{j,n}\Big\}}{\mathbb{E}\Big\{Y_n U_{1,n}\Big\}}, \quad j = 2, \cdots, d. \tag{27}$$

This yields the following estimate [12] of γ_j^*:

$$\frac{\hat{\gamma}_j}{\gamma_1^*} = \frac{\sigma_1^2}{\sigma_j^2} \frac{N^{-1} \sum_{t=1}^{N} Y_t U_{j,t}}{N^{-1} \sum_{t=1}^{N} Y_t U_{1,t}}, \;\; j = 2, \cdots, d. \tag{28}$$

The simulation studies will focus mostly on minimizing MSE approach in estimation of the parametric part. Direct estimate using derivative method will also be examined as a comparison with the former approach.

3.2.2 Nonparametric Estimation

Estimation of $g^*(\cdot)$ is obtained by plug in the estimate $\hat{\gamma}$ into (25):

$$\hat{g}(w) = \hat{g}(w; \hat{\gamma}), \tag{29}$$

but using the whole data set rather than using a subset T_1. It can also be shown by similar procedures as in [12], [23] that as $N \to \infty$,

$$\hat{\gamma} \to \gamma^*, \;\; (\mathbf{P}) \tag{30}$$

and by continues mapping theorem, that $\hat{g}(w; \hat{\gamma}) - \hat{g}(w; \gamma^*) \to 0, \;\; (\mathbf{P})$. It can also be shown that $\hat{g}(w; \gamma^*) - g^*(w) \to 0, \;\; (\mathbf{P})$, thus

$$\begin{aligned} \hat{g}(w; \hat{\gamma}) - g^*(w) &= (\hat{g}(w; \hat{\gamma}) - \hat{g}(w; \gamma^*)) + (\hat{g}(w; \gamma^*) - g^*(w)) \\ &\to 0, \;\; (\mathbf{P}) \end{aligned} \tag{31}$$

3.2.3 Monte Carlo Evaluation of Indentification Algorithms

In order to evaluate estimation errors, L (e.g., $L = 100$) repetitions of the same experiment sets are simulated from the system.

Error for the evaluation of the quality of the estimate $\hat{\gamma}$ is measured by

$$\text{Err}(\hat{\gamma}) = \frac{1}{L}\sum_{t=1}^{L} ||\hat{\gamma}^{[t]} - \gamma^*||^2,$$

where $\hat{\gamma}^{[t]}$ is the value of the estimate $\hat{\gamma}$ obtained from the t-th training set.

Error for the evaluation of the quality of the nonparametric estimate $\hat{g}(w)$ is evaluated by the MISE (Mean Integrated Square Error). For a single training set we first obtain:

$$\text{ISE}(\hat{g}) \simeq \frac{1}{M}\sum_{i=1}^{M} \Big(g^*(W_i^{new}) - \hat{g}(W_i^{new})\Big)^2,$$

where M is the size of testing set of $\{(\mathbf{U}_1^{new}, Y_1^{new}), \cdots, (\mathbf{U}_M^{new}, Y_M^{new})\}$, and $W_i^{new} = \gamma^{*T}\mathbf{U}_i^{new}$. Next the MISE is obtained by:

$$\text{MISE}(\hat{g}) \simeq \frac{1}{L}\sum_{t=1}^{L} \text{ISE}^{[t]}(\hat{g}), \tag{32}$$

where $\text{ISE}^{[t]}(\hat{g})$ is the value of $\text{ISE}(\hat{g})$ obtained from the estimation of the t -th training set.

Note that it is reasonable and necessary to normalize γ^* in the model, e.g., $||\gamma^*|| = 1$ or $\gamma_1^* = 1$. We normalize such that $\gamma^* = 1$. For 2-dimensional input cases, we furthermore express γ as $\Big(\cos(\theta), \sin(\theta)\Big)$, and evaluate the error for this parametric part by:

$$\text{Err}(\hat{\theta}) = \frac{1}{L}\sum_{t=1}^{L} |\hat{\theta}^{[t]} - \theta^*|^2. \tag{33}$$

A practical question is how to select the kernel bandwidth in nonparametric estimation. It is reasonable to use different bandwidths, one for the preliminary estimate $\hat{q}(\cdot;\gamma)$, and the other for the final estimate $\hat{q}(\cdot)$. We denote these two values as h_1 and h_2.. The bandwidth selection, as well as other implementation issues, will be discussed in Section 3.4.

3.2.4 Auxiliary Proofs

The following proof can be also found in [12].

Proofs of (23) Since the normalization is $\lambda_0^* = 1$, the predictive output is given by:

$$\begin{aligned} \hat{Y}_n &= \sum_{i=0}^{\infty} \lambda_i V_{n-i}(\gamma) = \sum_{i=0}^{\infty} \lambda_i g(W_{n-i}(\gamma)), \\ &= g(W_n(\gamma)) + \xi_n, \end{aligned}$$

where $W_n(\gamma) = \gamma^T U_n$ is the projected input signal onto the direction defined by the vector $\gamma \in \mathbb{R}^d$, $\xi_n = \sum_{i=1}^{\infty} \lambda_i g(W_{n-i}(\gamma))$. Without loss of generality, we can assume that $\lambda_0 = 1$ and $\mathbb{E}\{g(W_n(\gamma))\} = 0$ for each admissible (γ, λ), therefore ξ_n can be treated as zero mean noise and independent of $W_n(\gamma)$, or independent of $\mathbf{U}_n$. Hence the criterion (22) becomes:

$$Q\Big(\lambda, \gamma, g(\cdot)\Big) = \mathbb{E}|Y_n - \xi_n - g(W_n(\gamma))|^2.$$

For a given (λ, γ) pair, the minimum of $Q(\lambda, \gamma, g(\cdot))$ with respect to $g(\cdot)$ is attained by the regression function

$$\begin{aligned} g(w;\gamma) &= \mathbb{E}\{Y_n - \xi_n | W_n(\gamma) = w\} = \mathbb{E}\{Y_n | W_n(\gamma) = w\} - \mathbb{E}(\xi_n) \\ &= \mathbb{E}\{Y_n | W_n(\gamma) = w\}, \end{aligned}$$

where independence of ξ_n and $W_n(\gamma)$ has been used, as well as the fact that $\mathbb{E}(\xi_n) = 0$. Note that the solution $g(\cdot;\gamma)$ is independent of λ.

Plugging $g(\cdot;\gamma)$ into $Q\Big(\lambda, \gamma, g(\cdot)\Big)$, we get

$$Q(\lambda, \gamma) = \mathbb{E}[\text{var}\{(Y_n - \tilde{\xi}_n) | W_n(\gamma)\}], \tag{34}$$

where $\tilde{\xi}_n$ is the version of ξ_n with $g(\cdot)$ replaced by $g(\cdot;\gamma)$. Then we will examine the minimum of $Q(\lambda, \gamma)$ with respect to λ. Note that

$$Y_n = m^*(\mathbf{U}_n) + \eta_n + \varepsilon_n,$$

where $\eta_n = \sum_{i=1}^{\infty} \lambda_i^* m^*(\mathbf{U}_{n-i})$. Since $Y_n - \tilde{\xi}_n = m^*(\mathbf{U}_n) + \varepsilon_n + \eta_n - \tilde{\xi}_n$, thus the conditional variance term in (34) becomes:

$$\text{var}\{(Y_n - \tilde{\xi}_n)|W_n(\gamma)\} = \text{var}(m^*(\mathbf{U}_n)|W_n(\gamma)) + \text{var}(\varepsilon_n) + \text{var}\{\eta_n - \tilde{\xi}_n\}. \tag{35}$$

The only term in (35) depending on λ is the last term. Note that

$$\begin{aligned} \eta_n - \tilde{\xi}_n &= \sum_{i=1}^{\infty} \{\lambda_i^* m^*(\mathbf{U}_{n-i}) - \lambda_i g(W_{n-i}(\gamma);\gamma)\} \\ &= \sum_{i=1}^{\infty} \lambda_i^* \Big\{ m^*(\mathbf{U}_{n-i}) - g(W_{n-i}(\gamma);\gamma) \Big\} + \sum_{i=1}^{\infty} (\lambda_i^* - \lambda_i) g(W_{n-i}(\gamma);\gamma). \end{aligned}$$

As a result, we have the last term in (35) is given by:

$$\sum_{i=1}^{\infty} \lambda_i^{*2} \text{var}\Big\{ m^*(\mathbf{U}_{n-i}) - g(W_{n-i}(\gamma);\gamma) \Big\} + \sum_{i=1}^{\infty} (\lambda_i^* - \lambda_i)^2 \text{var}\Big(g(W_{n-i}(\gamma);\gamma) \Big).$$

This indicates that $Q(\lambda, \gamma)$ in (34) is minimized by $\lambda = \lambda^*$. Thus (23) has been proved.

Plugging $\lambda = \lambda^*$ into (34), the corresponding error has the following form

$$Q(\lambda^*, \gamma) = \mathbb{E}[\text{var}(m^*(\mathbf{U}_n)|W_n(\gamma))] + \text{var}(\varepsilon_n) + \sum_{i=1}^{\infty} \lambda_i^{*2} \text{var}\{m * (\mathbf{U}_{n-i}) - g(W_{n-i}(\gamma);\gamma\}.$$

This defines the profile risk $Q(\gamma)$ for the projection parameter γ. It is worth mentioning that $Q(\gamma)$ can also be written int he following form:

$$Q(\gamma) = \mathbb{E}\Big\{ \Big(Y_n - \sum_{i=0}^{\infty} \lambda_i^* g(W_{n-i}(\gamma);\gamma) \Big)^2 \Big\}, \tag{36}$$

where $g(w;\gamma) = \mathbb{E}\{Y_n | \gamma^T \mathbf{U}_n = w\}$.

The minimizer of $Q(\gamma)$ gives the optimal value of γ,i.e., we define

$$\gamma^* = \arg\min_{\gamma \in \Gamma} Q(\gamma). \tag{37}$$

As the solution for the optimal nonlinearity $g(w;\gamma)$ is coupled with the projection parameter γ, we finally obtain the best nonlinearity in the model is characterized by

$$g^*(\cdot) = g(w; \gamma^*).$$

The above considerations indicate:

1. The minimum of $Q(\lambda, \gamma, g(\cdot))$ with respect to λ is λ^*, the impulse response of the true system. This holds regardless of $g(\cdot)$ or γ.

2. The minimization of $Q(\lambda, \gamma, g(\cdot))$ with respect to $g(\cdot)$ for a given (λ, γ) is given by the regression function

Proofs of (26) and (27) First, we need the following Lemma (Theorem 14.7 from [12]): Let $(\mathbf{U}, Y) \in \mathbb{R}^d \times \mathbb{R}$ be a pair of random vectors such that $\mathbf{U}$ has a density $f(\cdot)$ defined on the set $S \subseteq \mathbb{R}^d$. Suppose that $f(\cdot)$ has a continuous derivative and $f(\cdot)$ is zero on the boundary ∂S of S. Assume that the regression $M(\mathbf{u}) = \mathbb{E}\{Y|\mathbf{U} = \mathbf{u}\}$ has a derivative $\partial M(\mathbf{u})/\partial \mathbf{u}$. Then, we have,

$$\mathbb{E}(\partial M(\mathbf{U})/\partial \mathbf{u}) = -\mathbb{E}\Big\{Y \frac{\partial f(\mathbf{U})/\partial \mathbf{u}}{f(\mathbf{U})}\Big\} \tag{38}$$

Note this is a direct consequence of integration by parts:

$$\begin{aligned}
\mathbb{E}(\partial M(\mathbf{U})/\partial \mathbf{u}) &= \int_S \frac{\partial M(\mathbf{u})}{\partial \mathbf{u}} f(\mathbf{u}) d\mathbf{u} \\
&= M(\mathbf{u}) f(\mathbf{u})|_{\partial S} - \int_S M(\mathbf{u})(\partial f(\mathbf{u})/\partial \mathbf{u}) d\mathbf{u} \\
&= -\int_S M(\mathbf{u})(\partial f(\mathbf{u})/\partial \mathbf{u}) d\mathbf{u} \\
&= -\int_S M(\mathbf{u}) \frac{\partial f(\mathbf{u})/\partial \mathbf{u}}{f(\mathbf{u})} f(\mathbf{u}) d\mathbf{u} \\
&= -\mathbb{E}\Big\{Y \frac{\partial f(\mathbf{U})/\partial \mathbf{u}}{f(\mathbf{U})}\Big\}
\end{aligned}$$

For the semiparametric Hammerstein system, we have $Y_n = \sum_{j=0}^{\infty} \lambda_j^* m^*(\mathbf{U}_{n-j}) + \varepsilon_n$, where $m^*(\mathbf{u}) = g^*(\gamma^{*T}\mathbf{u})$, and due to $m^*(\mathbf{u}) = \mathbb{E}(Y_n|\mathbf{U}_n = \mathbf{u})$, we have:

$$\mathbb{E}(\partial m^*(\mathbf{U_n})/\partial \mathbf{u}) = \gamma^* \mathbb{E}(g^{*(1)}(W_n)),$$

where $W_n = \gamma^{*T}\mathbf{U}_n$. By virtue of (38), we get:

$$\gamma^* \mathbb{E}(g^{*(1)}(W_n)) = -\mathbb{E}\Big\{Y_n \frac{\partial f_{\mathbf{U}}(\mathbf{U}_n)/\partial \mathbf{u}}{f_{\mathbf{U}}(\mathbf{U}_n)}\Big\}.$$

Hence,

$$\frac{\gamma_j^*}{\gamma_1^*} = \frac{\mathbb{E}\Big\{Y_n \frac{\partial f_{\mathbf{U}}(\mathbf{U}_n)/\partial u_j}{f_{\mathbf{U}}(\mathbf{U}_n)}\Big\}}{\mathbb{E}\Big\{Y_n \frac{\partial f_{\mathbf{U}}(\mathbf{U}_n)/\partial u_1}{f_{\mathbf{U}}(\mathbf{U}_n)}\Big\}},$$

and (26) has been proved.

Notice that if $\{\mathbf{U}_n\} \sim N_d(\mathbf{0}, \Sigma)$, then

$$\frac{\partial f_{\mathbf{U}}(\mathbf{u})/\partial \mathbf{u}}{f_{\mathbf{U}}(\mathbf{u})} = -\Sigma^{-1}\mathbf{u},$$

thus

$$\frac{\gamma_j^*}{\gamma_1^*} = \frac{\mathbb{E}\{Y_n(\Sigma^{-1}\mathbf{U}_n)_j\}}{\mathbb{E}\{Y_n(\Sigma^{-1}\mathbf{U}_n)_1\}},$$

where $(\Sigma^{-1}\mathbf{U}_n)_j$ denotes the j-th coordinate of the vector $(\Sigma^{-1}\mathbf{U}_n)$. This confirms (27).

3.3 Approximation of a MISO Hammerstein System by a Semiparametric Model

In practice, it is of interest to find the best semiparametric representation for a given MISO Hammerstein model. Namely, we want to use a semiparametric model (Fig. 10) to approximate a MISO Hammerstein model (Fig. 9), and wish to discover the optimal parametric part $\gamma = (\gamma_1, \gamma_2, \cdots, \gamma_d)^T \in \mathbb{R}^d$ and nonparametric part $g(\cdot)$ which minimize the mean squared error (22). For the system in Fig. 9, it shows in Section 3.2.4 that $Q(\gamma, g(\cdot))$ is minimized with respect to $g(\cdot)$ for a given γ by the following regression function

$$g(w;\gamma) = \mathbb{E}(Y_n|\gamma^T\mathbf{U}_n = w) = \mathbb{E}(m^*(\mathbf{U}_n)|\gamma^T\mathbf{U}_n = w). \tag{39}$$

Note second equality in (39) holds because of (20). Plugging the solution in (39) into $Q(\gamma, g(\cdot))$, we can obtain the general formula (36) for the smallest projection error for a given γThe minimizer of $Q(\gamma)$ in (36) defines the best possible γ and is denoted as γ^* (3.2.1). Note that γ needs to be specified with normalization constraints, i.e., $||\gamma|| = 1$. The optimal nonlinearity $g^*(w)$ is then defined as $g^*(w) = g(w;\gamma^*)$, and the corresponding minimal error of the proposed approximation is $Q(\gamma^*)$.

In the simulation studies, an example of semiparametric approximation will be examined as as an illustration.

3.4 Simulation Studies

In the following simulation studies, we are going to examine the identification of semiparametric model through the methods described in the previous section. Section 3.4.1-3.4.8 will be focused on the case that the true MISO Hammerstein system can be correctly represented by the semiparametric model, as the system in Fig. 11. Section 3.4.1-3.4.4 will be focused on semiparametric estimation under different nonlinear function, when different length of training data is available, when the input is of different dimensions, as well as when the input is correlated in its different dimensions. The kernel estimate bandwidth h_1 will be treated as another variables besides "θ" and will be determined together with obtaining $\hat{\theta}$ when minimizing the criterion function (24) in each run of the simulation. The selection of kernel bandwidth h_2 in obtaining $\hat{g}(\cdot)$ will be discussed and compared in the simulation studies in Section 3.4.1. In Section 3.4.5, the asymptotic distribution of such estimate will be examined by simulations, and visual plot about nonparametric estimate "$\hat{g}(\cdot)$" will also be shown with the true characteristics $g^*(\cdot)$ as a comparison. Section 3.4.6 will be about selection of kernel bandwidth h_1 and h_2. It shows another option: use fixed values beforehand for h_1, and h_2, and this approach is also feasible due to its performance is close the other approach used in Section 3.4.1-3.4.4. In Section 3.4.6, simulations will also examine another question: How performance in semiparametric estimation will change if a wrong kernel size is used. This could be due to practial reasons such as the width of grids that optimization criterion function is evaluating. Section 3.4.6 will also examine how the accuracy of estimating the parametric part will influence the estimation of the non-parametric part $\hat{g}(\cdot)$. Section 3.4.7 will conduct simulations to see how the ratio between the two subsets in partition data in estimating γ^* will have influence on the final performance. It will also examine another re-sampling technique: leave-one-out. Furthermore, the minimization of MSE approach in estimation of γ^* will be compared in Section 3.4.8 with the derivative method, which is a direct approach in obtaining the formula of solution described in Section 3.2.1. Finally, the issue that approximating a true MISO Hammerstein model by a semiparametric one will be shown in Section 3.4.9. In this case, "γ^*" and "$g^*(\cdot)$" are first determined by calculations. Then simulations will be performed on how close semiparametric estimate can represent the true nonlinear characteristic in the MISO model.

3.4.1 Estimation Error vs Smoothness of Nonparametric Characteristic

The shape of nonlinear function usually have a great influence on the performance of semi-parametric estimation, as well as the optimal kernel bandwidth. In the first simulation example, identification of semiparametric Hammerstein model is examined in the context of identifying different nonlinear functions. We use $g^*(w) = \alpha \arctan(\beta w)$. The parameter β here reflects how "difficult" the nonlinear function is. i.e., within the same interval symmetric to the origin, the smaller β is, the closer $g^*(\cdot)$ is to a linear function, while the greater β is, the closer $g^*(\cdot)$ is to a function that has a jump at $w = 0$. See Fig. 14 for this series of functions. Note α here is to ensure $\mathbb{E}\Big(\arctan(\beta W)\Big)^2$ is invariant for different selections of β, and $\alpha(\beta = 2) = 0.7$.

For all different nonparametric functions we examined, we use the same conditions: In-

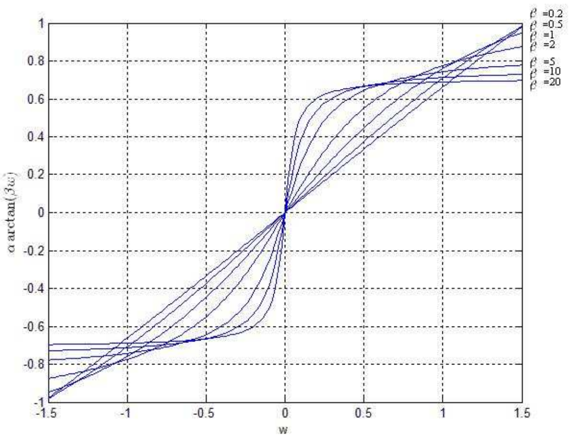


Figure 14: Plot of $g^*(w) = \alpha \arctan(\beta w)$ for different β.

put is a 2-dimensional i.i.d. Gaussian process $\mathbf{U} \sim \mathbf{N}_2\Big(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\Big)$, $\gamma^* = \Big(\cos(\theta^*),$ $\sin(\theta^*)\Big)^T$, where $\theta^* = \pi/4$, and we use $\Big(\cos(\hat{\theta}), \sin(\hat{\theta})\Big)^T$ to estimate the parametric part. Linear part is a FIR(3) filter: $\Lambda = [1, -0.8, 0.6, 0.4]$, i.e., $G_n = V_n - 0.8V_{n-1} + 0.6V_{n-2} - 0.4V_{n-3}$ in Fig. 9. And the noise is $\varepsilon_n \sim$ i.i.d. $N(0, 0.1)$, which corresponds to a 9.57dB SNR in the system despite of different β. The length of data set is 150, and we use approximately 55% of the whole data set to obtain $\hat{g}(\cdot;\gamma)$. The kernel bandwidth h_1 here is used as the value that minimize the contrast function in (24). Denote its value by $\hat{h}_1$. Repeat the same experiment for $L = 100$ times, and use (33) to evaluate the average identification error of the parametric part and (32) to evaluate the average error in the nonparametric part. We show the simulation results in the following Table 1.

Not surprisingly, we notice that as the nonlinear function becomes steeper (larger value of β), the identification error for the parametric part increases, and the optimal bandwidth $\hat{h}_1$ becomes smaller. For $\beta = 2$, Err($\hat{\theta}$) is 0.002631 on average. This is equivalent to an average 2.94 degrees error from the true value θ^*.

Once $\hat{h}_1$ is determined by optimization, there exists several different ways to select h_2. It makes sense to use some sub-sampling scheme to select $\hat{h}_2$. On the other hand, [24] shows that the optimal bandwidth of kernel density is:

$$h = (\frac{C}{N})^{-1/5}, \tag{40}$$

where C is a constant that depends on the nonlinear function $m^*(\cdot)$ as well as the kernel function $K(\cdot)$.

Due to the relationship in (40), it is also reasonable to select $\hat{h}_2 = c \cdot \hat{h}_1$, where c is a constant determined by the ratio of length of T_1 over the length of the whole data. So we compare the following three strategies:

Case 1. The value $\hat{h}_2$ is selected by the leave-(p+1)-out method in order to ensure the independence of the resampled data. Hence, the bandwidth for specifying the estimate $\hat{g}(\cdot)$ is obtained by minimizing the following criterion

$$\hat{Q}_{CV}(h_2) = \frac{1}{N-p} \sum_{n=p+1}^{N} \Big(Y_n - \sum_{i=0}^{p} \hat{\lambda}_i \hat{g}_{-[n-i,p+1])}(\hat{W}_{n-i}) \Big)^2,$$

where $\hat{W}_i = \hat{\gamma}^T U_i$, and $\hat{g}_{-[n-i,p+1]}(\cdot)$ is the version of the kernel estimate $\hat{g}(\cdot)$ (with the bandwidth h_2) computed from all the training points except

$$\{(\mathbf{U}_{n-i}, Y_{n-i}), (\mathbf{U}_{n-i-1}, Y_{n-i-1}), \cdots, (\mathbf{U}_{n-i-p}, Y_{n-i-p})\}.$$

β	$\hat{h}_1$	Err($\hat{\theta}$)
0.2	5.522 (1.868)	0.001127 (0.002080)
0.5	4.073 (2.036)	0.001312 (0.004081)
1	2.809 (1.677)	0.002038 (0.003514)
2	1.681 (1.153)	0.002631 (0.006035)
5	1.080 (0.603)	0.003249 (0.005718)
10	0.849 (0.399)	0.003546 (0.005377)
20	0.726 (0.173)	0.004474 (0.008064)

Table 1: Estimation errors and optimal kernel bandwidth in parametric part of the semi-parametric model in the stated experiment for different values of β. The mean as well as standard deviation of the error and bandwidth are shown.

Case 2. Use the kernel window size the same as the one in the first stage of the semiparametric algorithm, $\hat{h}_2 = \hat{h}_1$. This serves as the comparing case for not distinguishing between the two bandwidths in the process of semiparametric identification.

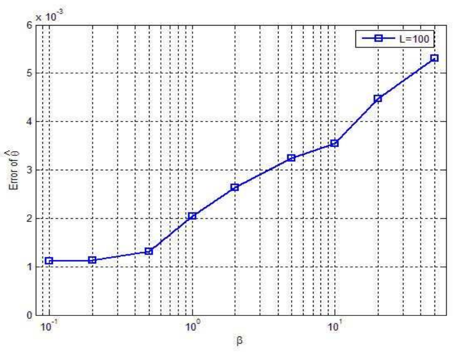
Case 3. Use $\hat{h}_2$ proportional to $\hat{h}_1$, i.e., $\hat{h}_2 = c \cdot \hat{h}_1$, where $c = (N/N_1)^{-\frac{1}{5}}$, and N_1 is the length of the training set T_1.

These different approaches are examined by simulations. $M = 200$ observations are generated as the testing set in each run of simulation of obtain an approximation of MISE$(\hat{g})$. Comparisons are shown in Table 2. Clearly the identification in Case 2 is not as good as in Case 3, this shows the necessity of making distinction between the two bandwidths in the two stages of nonparametric estimation. Comparing Case 1 and Case 3, we see when the nonparametric function $g^*(\cdot)$ is not too flat ($\beta > 1$), re-sampling (Case 1) will always lead to much higher value of $\hat{h}_2$, and have higher nonparametric estimation error. Conclusion can be made that when $g^*(\cdot)$ is not too close to linear, using the bandwidth selected in calculating $\hat{g}(\cdot;\gamma)$ multiplied by a known factor c is a better option compared with applying re-sampling again in selecting h_2 the final stage of nonparametric estimation. In most of the following sections the kernel bandwidth of h_2 is selected in this way. The estimation error for identification of parametric and nonparametric parts are also shown in Fig. 15.

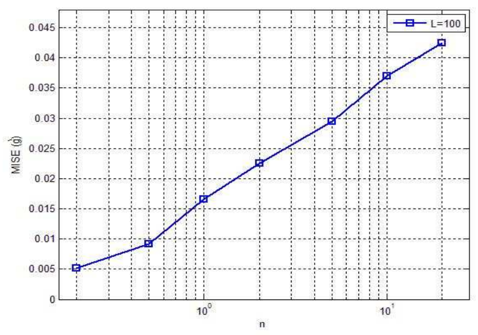
β	$\hat{h}_2^{[\text{Case 1}]}$	MISE$(\hat{g}(\cdot))^{[\text{Case 1}]}$	MISE$(\hat{g}(\cdot))^{[\text{Case 2}]}$	MISE$(\hat{g}(\cdot))^{[\text{Case 3}]}$
0.2	5.3295 (1.5275)	0.004514 (0.004520)	0.004721 (0.006502)	0.005224 (0.008485)
0.5	4.9650 (1.1418)	0.008909 (0.007909)	0.009578 (0.011578)	0.009228 (0.007959)
1	4.1740 (0.9929)	0.019215 (0.008037)	0.016471 (0.012053)	0.016533 (0.012840)
2	3.0300 (1.1704)	0.030738 (0.013092)	0.022295 (0.012897)	0.022536 (0.013521)
5	1.2055 (0.8695)	0.031996 (0.016693)	0.030481 (0.019466)	0.029457 (0.016275)
10	0.8160 (0.4758)	0.037975 (0.019514)	0.037981 (0.019530)	0.037048 (0.019300)
20	0.6990 (0.4319)	0.044579 (0.016798)	0.043602 (0.014725)	0.042455 (0.015170)

Table 2: Performance of nonparametric identification using different strategies in selecting h_2.

It is also worth mentioning that estimation of errors and kernel bandwidths in Table 1



(a)



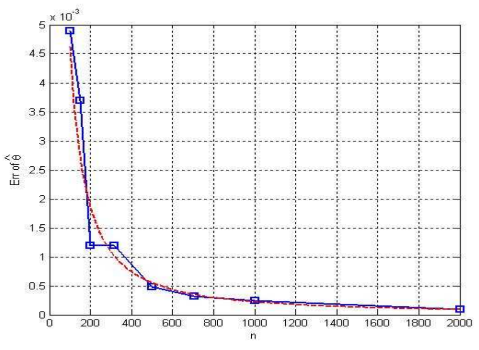
(b)

Figure 15: Semiparametric estimation error vs β. (a) Err$(\hat{\theta})$ vs β, (b) MISE$(\hat{g})$ vs β.

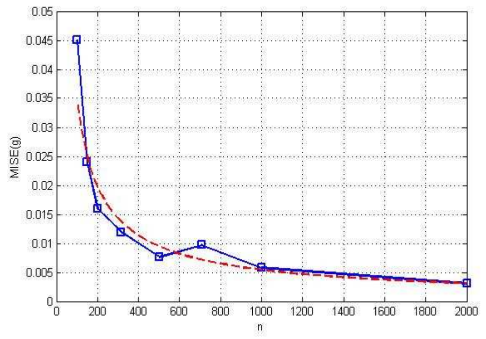
and Table 2 all show values with a great variance. This is perhaps the nature of semiparametric identification.

3.4.2 Estimation Error vs Data Size

It is of interest to see how semiparametric estimation performs under different length of training data set. Again, Input $\mathbf{U} \sim$ i.i.d. $\mathbf{N}_2\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$, $\gamma^* = \left(\cos(\theta^*), \sin(\theta^*)\right)^T$, where $\theta^* = \pi/4$, and we use $\left(\cos(\hat{\theta}), \sin(\hat{\theta})\right)^T$ to estimate the parametric part. $g^*(w) = 0.7\arctan(2w)$. Linear part is a FIR(3): $\Lambda = [1, -0.8, 0.6, 0.4]$, and the noise is $\varepsilon_n \sim$ i.i.d. $N(0, 0.1)$, which corresponds to a 9.57dB SNR for the system. Use approximately 55% of the whole data set to obtain $\hat{g}(\cdot;\gamma)$. The kernel bandwidth h_1 is used as the value $\hat{h}_1$ that minimize the contrast function in (24). Use $\hat{h}_2 = c \cdot \hat{h}_1$ in obtaining $\hat{q}(\cdot)$. Examine identification error for different value of N, which is the length of observed data set. Repeat the same experiment for $L = 100$ times, and use (33) and (32) to evaluate average identification error in the parametric part and in the nonparametric part. The simulation result is in Fig. 16.



(a)



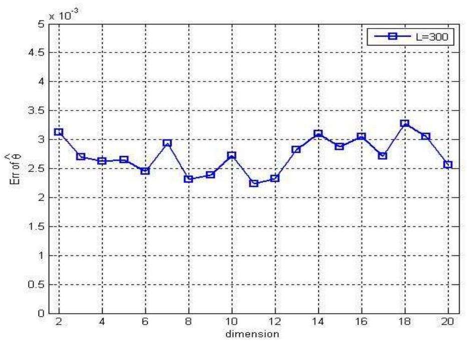
(b)

Figure 16: Semiparametric estimation error vs N. (a) Err$(\hat{\theta})$ vs N, (b) MISE$(\hat{g})$ vs N.

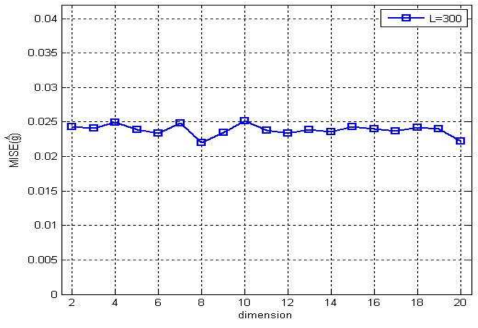
3.4.3 Estimation Error vs Dimension of Input

In this section we will examine the performance of semiparametric estimation for different dimensions of input data: $\mathbf{U} \sim$ i.i.d. $\mathbf{N}_d\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix} \right)$. $\gamma^* = \Big(\cos(\theta^*), \frac{1}{\sqrt{d-1}} \sin(\theta^*),$ $\cdots, \frac{1}{\sqrt{d-1}} \sin(\theta^*) \Big)^T$, with $\theta^* = \pi/4$, and we use $\Big(\cos(\hat{\theta}), \frac{1}{\sqrt{d-1}} \sin(\hat{\theta}), \cdots, \frac{1}{\sqrt{d-1}} \sin(\hat{\theta}) \Big)^T$ to estimate the parametric part. $g^*(w) = 0.7 \arctan(2w)$. Linear part is FIR(3): $\Lambda = [1, -0.8, 0.6, 0.4]$, and the noise is $\varepsilon_n \sim$ i.i.d. $N(0, 0.1)$, which corresponds to a 9.57dB SNR for the system. Use approximately 55% of the whole data set to obtain $\hat{g}(\cdot; \gamma)$. The kernel bandwidth h_1 is used as the value $\hat{h}_1$ that minimize the contrast function in 24, and use $\hat{h}_2 = c \cdot \hat{h}_1$ as the kernel bandwidth h_2. Examine identification error for different value of d. Repeat this for $L = 300$ times, and use (33) and (32) to evaluate average identification error in the parametric part and in the nonparametric part. The simulation result is in shown Fig. 17.

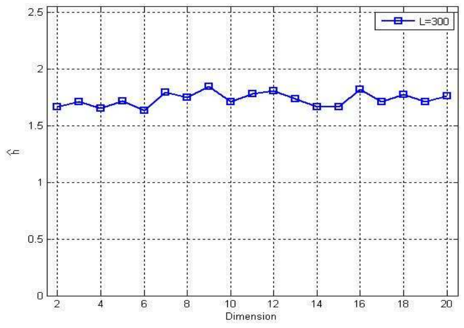
From Fig. 17(c), we see that optimal kernel window size is almost invariant for all the dimensionality. Fig. 17 shows that semiparametric estimation works equally well in higher dimensions compared with lower dimensions. If we conduct the same experiment, but applying nonparametric estimation directly to the input and output data, i.e., examine the regression of Y directly on multivariate input $\mathbf{U}$. Applying classical kernel estimation and we get Fig. 18. Note that this plot include the case that the input dimension is from 1



(a)



(b)



(c)

Figure 17: Semiparametric estimation error vs input dimensionality. (a) Err$(\hat{\theta})$ vs d, (b) MISE$(\hat{g})$ vs d, (c) Average optimal kernel bandwidth $\hat{h}_1$ vs d.

to 20, and $L = 200$ times of repetition is performed for calculating the average. This result should be compared with Fig. 17, and is an illustration for the "Curse of dimensionality".

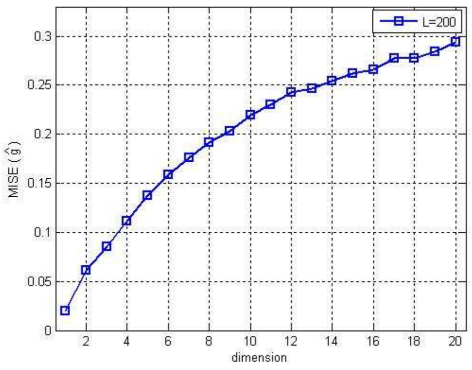
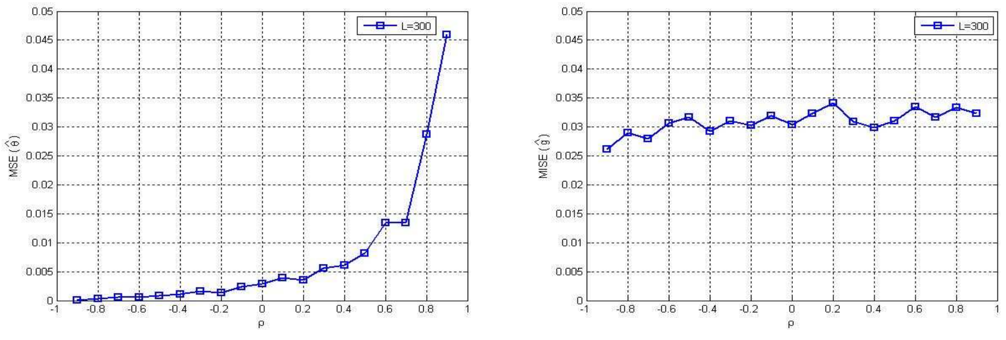


Figure 18: Direct nonparametric estimation for different dimensions of input. This reflects the so-called "Curse of Dimensionality", which is a typical shortcoming for nonparametric estimation.

3.4.4 Estimation Error vs Correlation of Input

In the next experiment, we are going to examine semiparametric estimation in the context of input signal having correlation between each of its dimensions. $\gamma^* = \Big(\cos(\theta^*), \sin(\theta^*)\Big)^T$, where $\theta^* = \pi/4$, and we use $\Big(\cos(\hat{\theta}), \sin(\hat{\theta})\Big)^T$ to estimate the parametric part. $g^*(w) = 0.7\arctan(2w)$. Linear part is a FIR(3): $\Lambda = [1, -0.8, 0.6, 0.4]$, and noise is $\varepsilon_n \sim$ i.i.d. $N(0, 0.1)$. which corresponds to a 9.57dB SNR for the system. Use approximately 55% of the whole data set to obtain $\hat{g}(\cdot;\gamma)$. The kernel bandwidth h_1 is used as the value that minimize the contrast function in (24), denote it by $\hat{h}_1$. Use $\hat{h}_2 = c \cdot \hat{h}_1$ as the kernel bandwidth in estimating $g^*(\cdot)$. The length of observed data set is $N = 150$. Let input **U** $\sim$ i.i.d. $\mathbf{N}_2\Big(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \alpha(\rho)\cdot\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}\Big)$. Note the term α is to make sure $\mathbb{E}(W)$ here is invariant for different ρ, and $\alpha(\rho = 0) = 1$. In this case the SNR of the system is always 9.57dB for different ρ. Examine semiparametric estimation for different choice of ρ, and repeat the same experiment for $L = 300$ times, and use (33) and (32) respectively to evaluate average identification error in the parametric part and in the nonparametric part. The simulation result is shown Fig. 16.



(a) (b)

Figure 19: Semiparametric estimation error vs input correlation. (a) Err$(\hat{\theta})$ vs ρ, (b) MISE$(\hat{(g(\cdot))})$ vs ρ.

It is not surprising to observe in Fig. 19 (b) that under different input correlation, as long as $\mathbb{E}(W)$ is invariant, then the nonparametric estimation has similar level of errors. On the other hand, Fig. 19(a) shows negative input correlation corresponds to smaller parametric errors in the semiparametric model. This is consistent with the intuition that negatively correlated input is able to make the parametric part more distinctive under this situation.

3.4.5 Distribution of Semiparametric Statistics

In Section 3.4.2, for $N = 150$, and $d = 2$, and repeat $L = 10000$ times for evaluation the error, if we plot the nonparametric estimate $\hat{g}(\cdot)$ and compare its shape with $g^*(\cdot)$, the result is shown in Fig. 20.

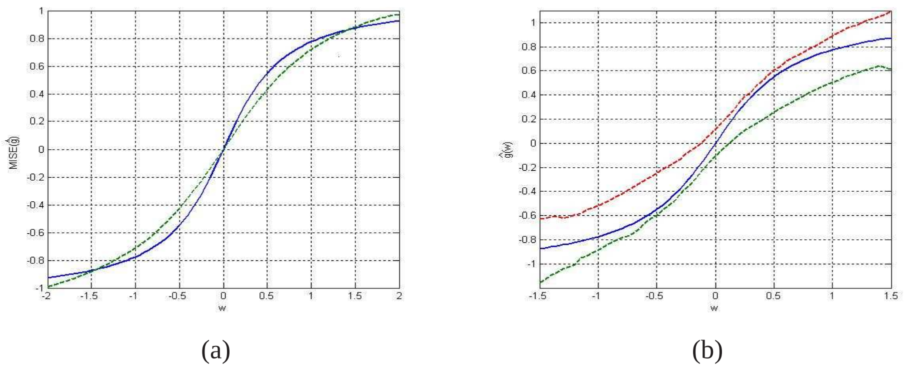
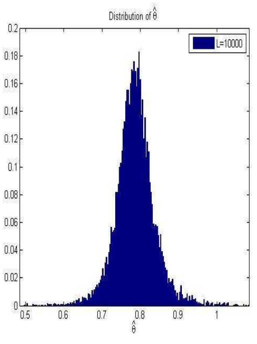


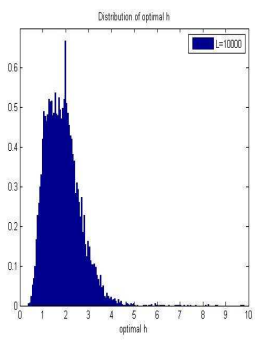
Figure 20: Estimation of nonparametric function $g^*(\cdot)$ in the semiparametric model. (a) The mean estimate (dash line) and the true function (solid line), (b) The 95% pointwise confidence interval of the estimate (dash line) and the true function (solid line).

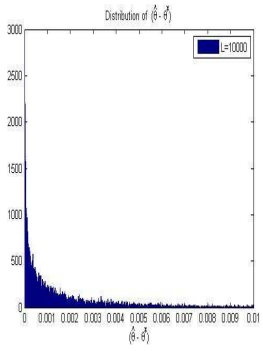
We then examine the distribution of $\hat{\theta}$, $\hat{h}_1$, $(\hat{\theta} - \theta^*)^2$ and $ISE(\hat{g}(\cdot))$. The histogram, as well as nonparametric estimation of these distributions are shown in Fig. 21. Not surprisingly, the distribution of $\hat{\theta}$ has a Gaussian shape, thus consistent with Central Limit Theorem. Note empirical mean for $\hat{\theta}$ is 0.786481 ($\theta^* = 0.785398$) and standard deviation for $\hat{\theta}$ is 0.055030.

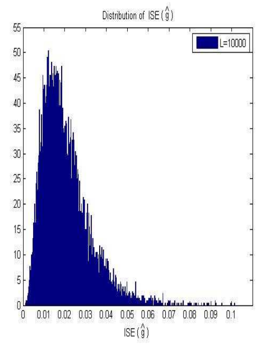
3.4.6 Estimation Error vs Kernel Bandwidth

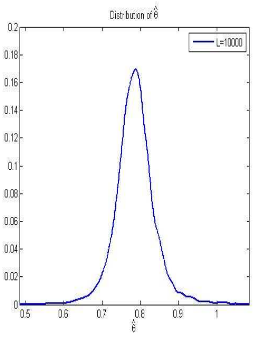
In previous simulation studies, we have been using flexible choice of kernel bandwidths, i.e., using the bandwidth that would lead to minimum contrast function in (24), thus selecting $\hat{h}_1$ the same time when obtaining $\hat{\theta}$, and use $\hat{h}_2 = c \cdot \hat{h}_1$ in obtaining $\hat{g}(\cdot)$. Thus the kernel bandwidths always differ from simulation to simulation even though the system

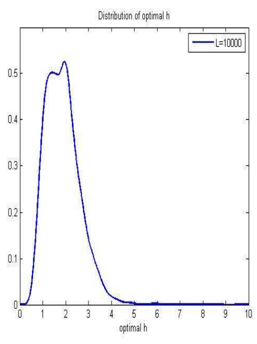

(a)

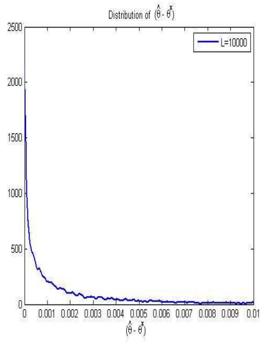

(b)


(c)


(d)


(e)


(f)


(g)

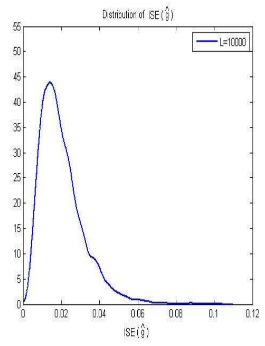
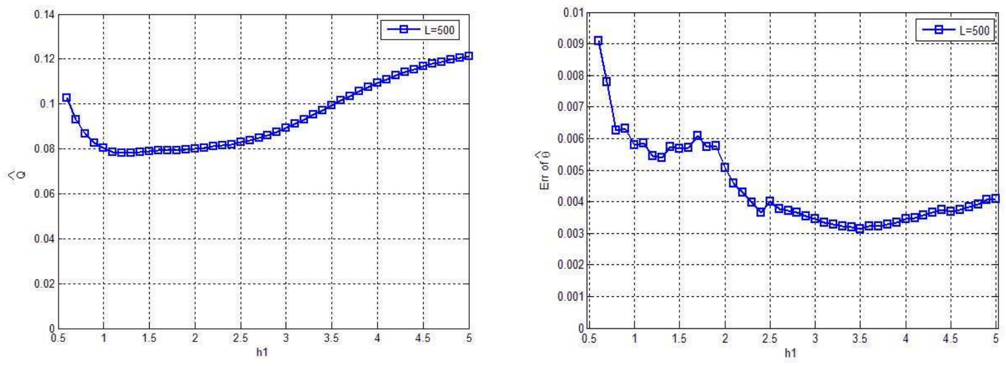

(h)

Figure 21: Distribution of semiparametric estimation statistics. (a) Histogram of $\hat{\theta}$, (b) Histogram of optimal $\hat{h}_1$, (c) Histogram of $(\hat{\theta} - \theta^*)^2$, (d) Histogram of $ISE(\hat{g})$, (e)-(h), nonparametric estimation for the distributions shown in (a)-(d).

is the same. Another approach is to fix kernel bandwidth beforehand, and use this fixed bandwidth in semiparametric estimation.

Given input $\mathbf{U} \sim$ i.i.d. $\mathbf{N}_2\Big(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\Big)$, $\gamma^* = \Big(\cos(\theta^*), \sin(\theta^*)\Big)^T$, where $\theta^* = \pi/4$, and we use $\Big(\cos(\hat{\theta}), \sin(\hat{\theta})\Big)^T$ to estimate the parametric part. $g^*(w) = 0.7\arctan(2w)$. Linear part is a FIR(3): $\Lambda = [1, -0.8, 0.6, 0.4]$, and the noise is $\varepsilon_n \sim$ i.i.d. $N(0, 0.1)$, which corresponds to a 9.57dB SNR for the system. Use approximately 55% of the whole data set to obtain $\hat{g}(\cdot; \gamma)$. The kernel bandwidth h_1 is fixed beforehand. Examine the error for parametric part under different value of h_1. Repeat the same experiment for $L = 500$ times, and use (33) to evaluate average identification error in the parametric part. The simulation result is in Fig. 22.

It shows that fixing $h_1 = 3.5$ in this case will lead to the smallest error. The average



(a) (b)

Figure 22: Estimation error in the parametric part using fixed kernel bandwidth. (a) The minimum contrast function vs h_1, (b) Err($\hat{\theta}$) vs h_1.

mean square error for $\hat{\theta}$ is 0.00314 (with standard deviation 0.006059). Compare this result with the flexible bandwidth case (Table 1) where average Err($\hat{\theta}$) is 0.00263 (and standard deviation 0.006035), we can conclude that fixing h_1 works relatively well. The curve near the minimum in Fig. 22(b) is rather flat, suggesting that using a wide range of values for bandwidth h_1 is relatively applicable. Therefore fixing bandwidth h_1 is another feasible option in semiparametric modeling.

Next, we come to the issue of selecting h_2. It might be reasonable to fix $h_2 = c \cdot h_1$, as (40) has suggested. In this case h_2 is specified by h_1. In previous simulation, we examine this approach, and compare MISE($\hat{g}$) under different selection of h_1. Fig. 23 shows this result. Although using kernel bandwidth $h_1 = 3.5$ in obtaining $\hat{g}(\cdot; \lambda)$ leads to averagely optimal estimate of $\hat{\theta}$, fixing kernel bandwidth $h_2 = c \cdot 3.5$ leads to poor result in estimating $\hat{g}(\cdot)$ from Fig. 23.

Another method is to use fixed h_2 totally independent of h_1. To the same $L = 500$ data sets, we use automatical selection of h_1 (as in Section 3.4.1-3.4.4) in obtaining $\hat{g}(\cdot; \lambda)$, and then use fixed h_2 in estimating $\hat{g}(\cdot)$. The result of this simulation is shown in Fig. 24.

This shows if we want to use fixed kernel window size, the value used in estimating $g^*(\cdot)$ should be 1.1 for training set of length 150. The mean and standard deviation for the mean integrated square error in this setting is 0.02264 and 0.01568. It is very close to the

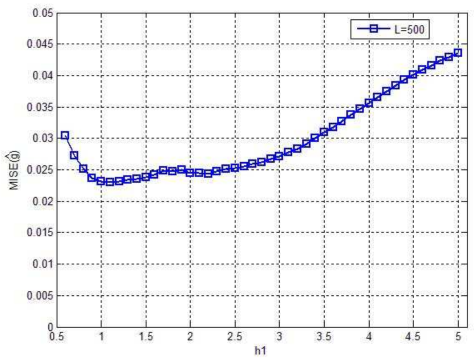


Figure 23: Estimation error in the nonparametric part using fixed kernel bandwidth $h_2 = c \cdot h_1$.

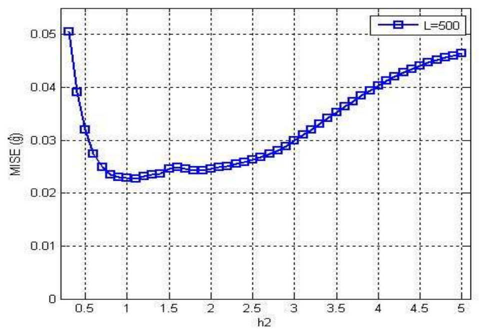


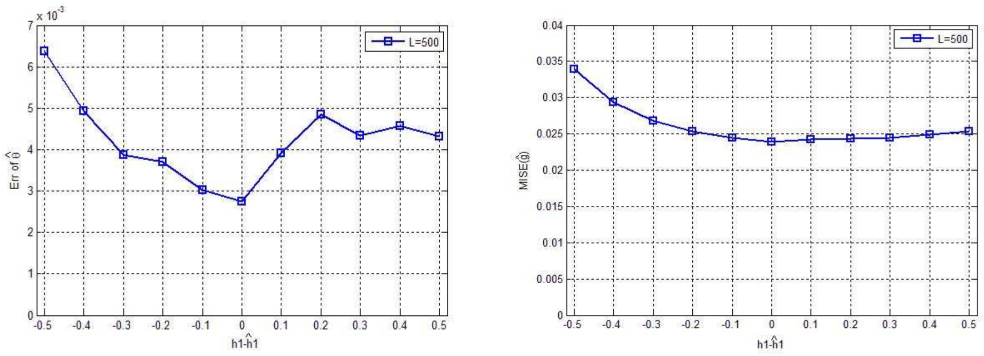
Figure 24: MISE$(\hat{g}(\cdot))$ vs h_2.

unfixed case in Table 2 (0.02254 and 0.01352).

Therefore we should select kernel bandwidth equal to 3.5 in estimating γ^*, and to be 1.1 in estimating $g^*(\cdot)$. For the training set with different length of data, we can modify these numbers by a multiplication factor accordingly.

Fig. 24 is when θ^* is estimated under proper selection of h_1, while for simulations in Fig. 23(b), it is usually not so. However, the estimate $\hat{g}(\cdot)$ have similar shapes in the two figures. Thus it suggests the estimation of $g^*(\cdot)$ and the estimation of θ^* are independent of each other, and under the circumstance even using improper kernel window size in estimating θ^*, properly selecting kernel window size in the second step would still give good estimate about the nonparametric function $g^*(\cdot)$.

On the other hand, in semiparametric estimation using flexible kernel bandwidth, misusing the kernel width (e.g., the error due to the length of grid we search in optimization) could lead to higher identification error. This is further examined in Fig. 25. In this simulation, the same $L = 500$ data sets are obtained. Let $\hat{h}_1$ is minimizer of contrast function, but a different value h_1 is used as kernel bandwidth in obtaining $\hat{g}(\cdot;\gamma)$, and $h_2 = c \cdot h_1$ is used in obtaining $\hat{g}(\cdot)$. It shows estimation of parametric part would be more subjective to kernel bandwidth, while estimation of nonparametric part is not so much sensitive.



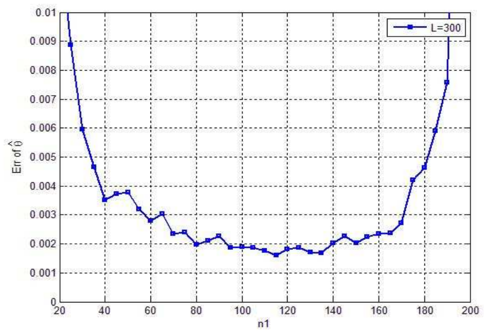
(a) (b)

Figure 25: Estimation error when sub-optimal kernel bandwidth is used. (a) Err($\hat{\theta}$) vs $(h_1 - \hat{h}_1)$, (b) MISE($\hat{g}(\cdot)$) vs $(h_1 - \hat{h}_1)$.

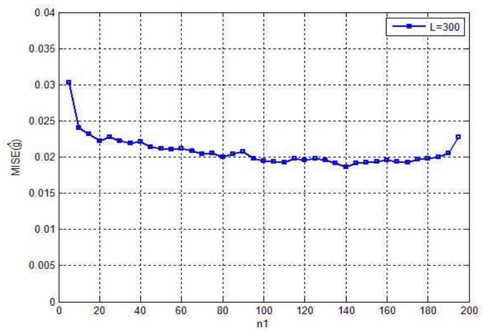
3.4.7 Discussions about Optimal Data Partition and Other Re-sampling Schemes

In the previous simulations, we have been using 55% of the total data as the set T_1 to calculate $\hat{g}(\cdot;\lambda)$. A question of interest is how much percent data should used in such re-sampling would give optimal identification result. The following experiment is using the same system and algorithms as in Section 3.4.2, with $N = 206$. And the simulation is to examine identification performance under different choices of N_1, the length of T_1. Note that $N = 206$ in this case is for the convenient that the total observations in set T_1 and T_2 are equal to 200. We then perform $L = 300$ repetitions of simulations to obtain average errors. The result is shown in Fig. 26.

From the simulation result, we observe using 55%-60% of whole data in obtaining $\hat{g}(w;\gamma)$ would lead to the most accurate estimate of γ^*. However, the length of this subset seems to have less significant influence in estimating $g^*(w)$, except for extreme cases.



(a)



(b)

Figure 26: System identification error vs length of T_1 subset. (a) Err$(\hat{\theta})$ vs N_1, (b) MISE$(\hat{g})$ vs N_1.

Besides, the minimum in Fig. 26(a) is rather flat, suggesting that a wide range of N_1/N ratio choices would be acceptable.

For the same system, if we observe data size $N = 150$, and use fixed kernel bandwidth $h_1 = 3.5$ and $h_2 = 1.1$, and to apply leave-one-out rather than partition re-sampling, we finally get Err$(\hat{\theta}) = 0.000621$, and MISE$(\hat{g}(\cdot)) = 0.021413$ in the simulation. Comparing this result with the partition method where Err$(\hat{\theta}^{[Partition]}) = 0.002631$ (corresponds to an average 1.428 degree of error) and MISE$(\hat{g}(\cdot)^{[Partition]}) = 0.022536$ (shown in Table 1 and Table 2), we see a significant improvement in estimation of parametric part, while estimation of nonparametric part still performs similarly. Leave-one-out method fully makes use of the total data set, so it is not surprising that it leads to better estimates. When only a small data set is available, leave-one-out is surely a powerful option. The similar performance in MISE$(\hat{g}(\cdot))$ again suggests re-sampling technique only have influence on parametric part, and identification of parametric part and nonparametric part are two relatively separated problems.

3.4.8 Comparison with Derivative Method in Parametric Inference

The previous simulations have been using minimization of MSE in estimating the parametric part of the model. If the parametric inference in Section 3.4.1 and Section 3.4.2 are performed through the derivative approach (See Section 3.2.1) rather than minimizing MSE, we get Fig. 27 and Fig. 28 respectively. Note that the constraint $||\gamma^*|| = 1$ is also used here, and we also use $(\cos(\hat{\theta}), \sin(\hat{\theta}))$ for representing $\hat{\gamma}$, as we did for the 2-dimensional cases in both Section 3.4.1 and Section 3.4.2, and use (33) to represent the estimation error for the parametric part.

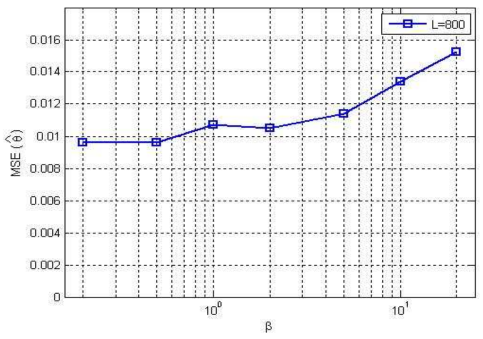


Figure 27: Err$(\hat{\theta})$ vs β using derivative approach in parametric inference.

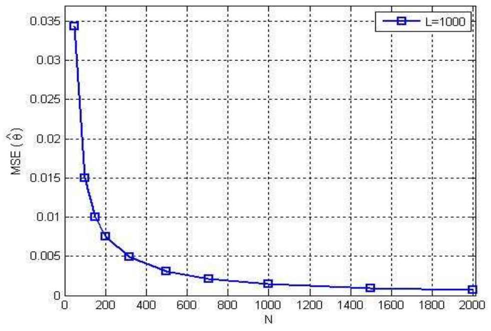


Figure 28: Err$(\hat{\theta})$ vs N using derivative approach in parametric inference.

Comparing with Section 3.4.1 and Section 3.4.2, we see that the prediction error is at least 3 times larger by applying derivative approach. Therefore, the derivative method, although explicit in solutions, does not work as good as minimizing MSE approach. However, due to its simplicity in application, it can always be applied as a first approach in semiparametric estimation, and its solution could be applied as an initial search position for applying the other method.

If we examine apply semiparametric estimation for different dimension of input, applying derivative approach to the system in Section 3.4.3, and use $\mathbb{E}(||\hat{\gamma} - \gamma^*||^2)$ as the indicator for estimation error, we get Fig. 29.

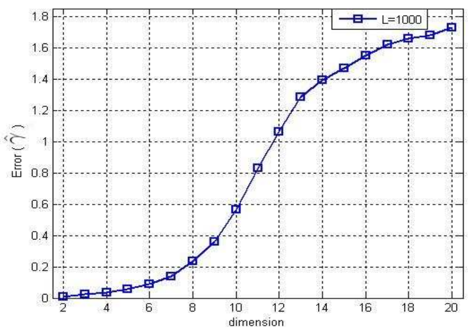


Figure 29: Err$(\hat{\gamma})$ vs dimension using derivative approach in parametric inference.

3.4.9 Semiparametric Approximation of a MISO Hammerstein System

In this section, we are going to examine the validity of applying semiparametric model to MISO Hammersteim systems. Suppose we observe data from a 2-dimensional Hammerstein system with the following nonlinearities:

$$m^*(\mathbf{U}_n) = \sigma(U_{n,1} + 3U_{n,2}) + \sigma(3U_{n,1} + U_{n,2}) - 1$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is the so-called sigmoid function, and $\mathbf{U}_n = (U_{n,1}, U_{n,2})^T$, $n = 1, \cdots$. The plot of this 2-d nonlinear function is shown in Fig. 30.

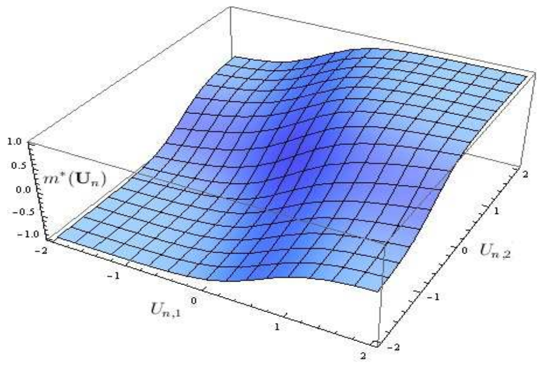
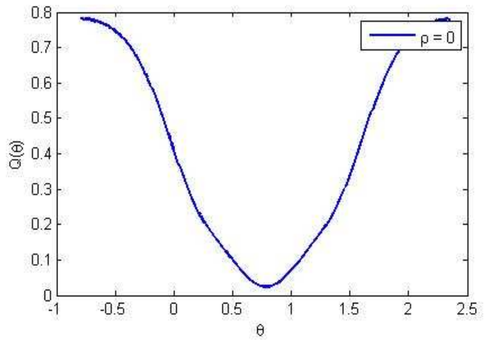


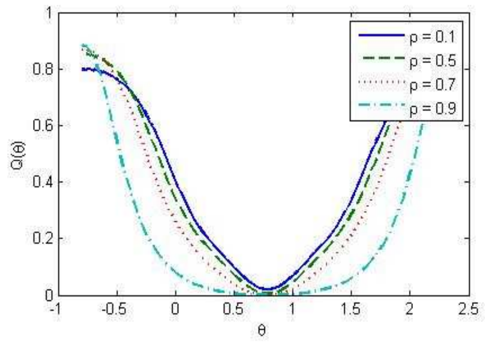
Figure 30: Err$(\hat{\gamma})$ vs dimension using derivative approach in parametric inference.

We notice that $m^*(\cdot)$ qualify $\mathbb{E}(m^*(\mathbf{U}_n)) = 0$. In order to determine the optimal parametric characteristic γ, we need to evaluate $Q(\gamma)$ in (36). Under the normalization

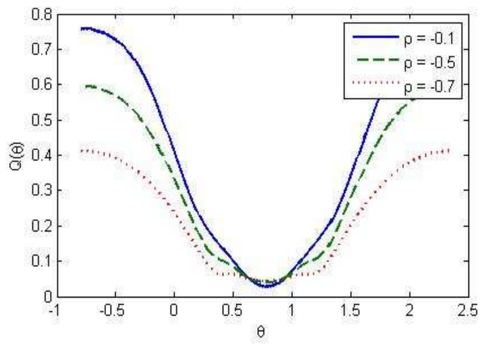
$||\gamma|| = 1$, we express the parametric part by $\gamma = (\cos(\theta), \sin(\theta))^T$. For input $\mathbf{U} \sim$ i.i.d. $\mathbf{N}_2\Big(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \Big)$ then the simulation of $Q(\theta)$ vs θ according to (36) under different input correlation ρ is shown in Fig. 31.



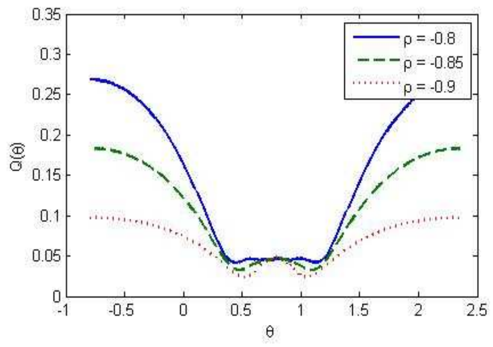
(a)



(b)



(c)

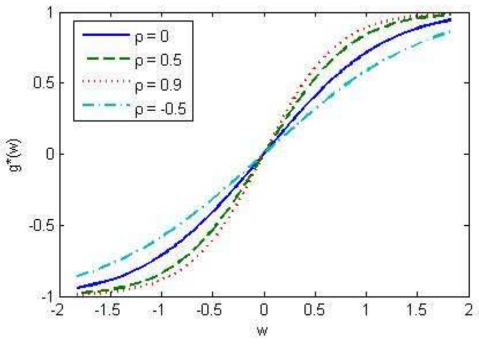


(d)

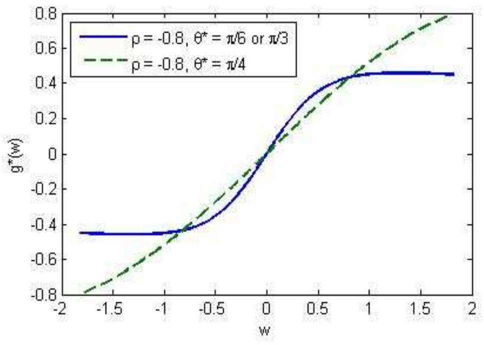
Figure 31: The smallest projection error $Q(\theta)$ vs θ, given different input correlation ρ. (a) $\rho = 0$, (b) $\rho = 0.1, 0.5, 0.7, 0.9$, (c) $\rho = -0.1, -0.5, -0.7$, (d) $\rho = -0.8, -0.85, -0.9$.

Fig. 31(a)&(b) shows that for $\rho \geq 0$, the approximation error function has its minimum at $\theta = \pi/4$. This means the parametric part in semiparametric model should have the characteristic $\theta^* = \pi/4$, i.e., $\gamma^* = (\sqrt{2}/2, \sqrt{2}/2)^T$. For negative values of correlation input ρ (Fig. 31(c)), it seems that for $\rho \geq -0.7$, θ^* should still be selected as $\pi/4$. As ρ becomes closer to -0.8 (Fig. 31(d)), the minimum of $Q(\theta)$ becomes harder and harder to identify, and when $\rho = -0.8$ (Fig. 31(c)), it seems the region $\theta \in \{\theta : \pi/6 \leq \theta \leq \pi/3\}$ all corresponds to the minimum of $Q(\theta)$. This means reasonably, θ^* could be any value between $\pi/6$ and $\pi/3$, although the semiparametric model lost its uniqueness in identification, it on the other shows the approximation to the MISO Hammerstein model is relatively “easy” since as long as θ^* is within $[\pi/6, \pi/3]$, the approximation error is almost equally the smallest. Fig. 31(d) also shows that when $\rho < -0.8$, the global minimum of $Q(\theta)$ no longer appears at $\theta = \pi/4$, but at $\theta = \pi/6$ and $\theta = \pi/3$ respectively. Therefore θ^* should be equal to either $\pi/6$ or $\pi/3$ in this case.

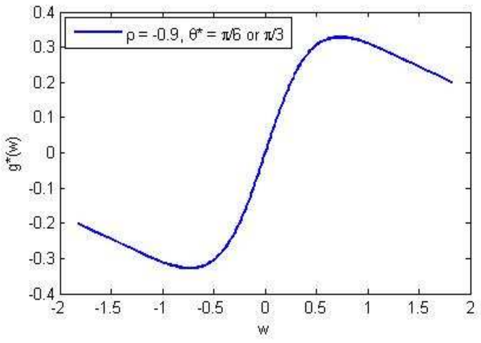
The nonlinear characteristic $g^*(\cdot)$ is obtained by (29). The result is shown in Fig. 32. For $\rho > -0.7$, $\theta^* = \pi/4$, and the shape of $g^*(\cdot)$ is shown in Fig. 32(a). For $\rho = -0.8$, θ^* does not have unique values, and Fig. 32(b) shows $g^*(\cdot)$ when θ^* is selected as $\pi/6$, $\pi/3$ and $\pi/4$. For $\rho = -0.9$, θ^* equals to either $\pi/6$ or $\pi/3$, and plug in the corresponding $g^*(\cdot)$ are the same for each case, and is shown in Fig. 32(c).



(a)



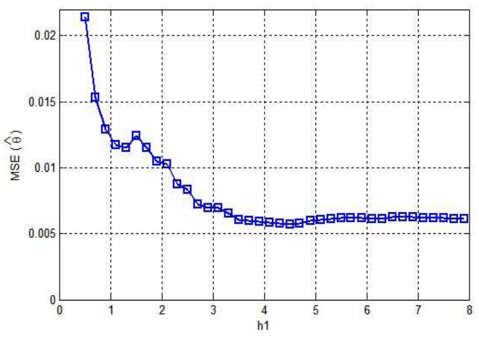
(b)



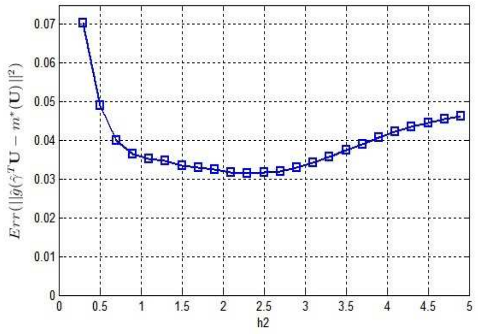
(c)

Figure 32: The nonlinear function $g^*(\cdot)$ in semiparametric approximation under different input correlation. (a) $\rho = 0, 0.5, 0.9, -0.5$, (b) $\rho = -0.8$, $\theta^* = \pi/6, \pi/3, \pi/4$, (c) $\rho = -0.9$, $\theta^* = \pi/6, \pi/3$.

We are interested in applying semiparametric model to the MISO Hammerstein system with with nonlinear subsystem $m^*(\mathbf{U}_n)$ and input signal described as above. Suppose the MISO Hammerstein model has linear subsystem FIR(3): $\Lambda = [1, -0.8, 0.6, -0.4]$, and $\rho = 0$ in the input covariance matrix, and the noise $\varepsilon_n \sim$ i.i.d. $N(0, 0.305^2)$ (10dB SNR for the system). Then we first examine the effect of kernel size in the semiparametric identification scheme described before when the training set is of length $N = 150$, then conduct a repetition of $L = 500$ different simulations. Fig. 33(a) shows average Err$(\hat{\theta})$ vs kernel size (denoted by h_1). We generate a testing data set containing $M = 200$ observations from the MISO Hammerstein model, and use $\frac{1}{L} \cdot \frac{1}{M} \sum_{i=1}^{M} |\hat{g}(\hat{\gamma}^T \mathbf{U}_i) - m^*(\mathbf{U}_i)|^2$ (where $\hat{\gamma} = (\cos(\theta), \sin(\theta))$) to evaluate the error in estimating the nonparametric characteristics $g^*(\cdot)$. Note this error is an estimate of $\mathbf{E}|\hat{g}(\hat{\gamma}^T \mathbf{U}) - m^*(\mathbf{U})|^2$, which is approximately $\mathbf{E}|\hat{g}(\hat{\gamma}^T \mathbf{U}) - g^*({\gamma^*}^T \mathbf{U})|^2$ plus the semiparametric projection error. Note since $\hat{g}(\hat{\gamma}^T \mathbf{U})$ is very close to $\hat{g}({\gamma^*}^T \mathbf{U})$ (see Section 3.4.6), $\mathbf{E}|\hat{g}(\hat{\gamma}^T \mathbf{U}) - m^*(\mathbf{U})|^2$ can be viewed as a statistics that reflects the error for $\hat{g}(\cdot)$. Then $\frac{1}{L} \cdot \frac{1}{M} \sum_{i=1}^{M} |\hat{g}(\hat{\gamma}^T \mathbf{U}_i) - m^*(\mathbf{U}_i)|^2$ vs kernel window size (denoted by h_2) is shown in Fig. 33(b).



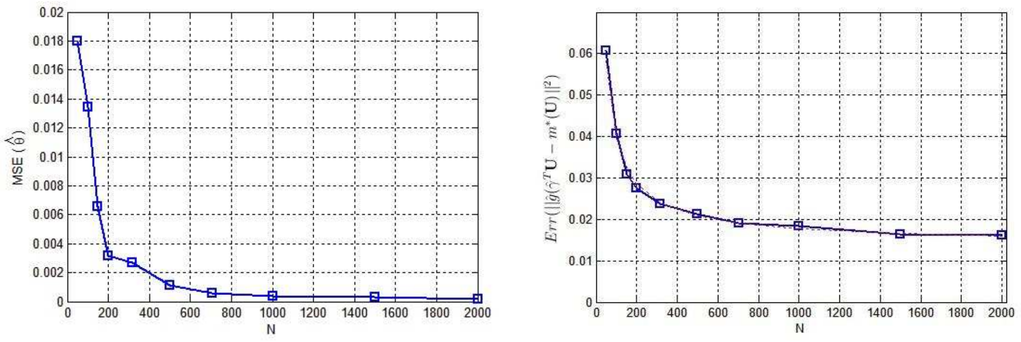
(a)



(b)

Figure 33: Semiparametric identification error vs kernel size. (a) Err$(\hat{\theta})$ vs h_1, (b) Err$(|\hat{g}(\hat{\gamma}^T \mathbf{U}) - m^*(\mathbf{U})|^2)$ vs h_2.

From Fig. 33, we see that the optimal kernel length for this problem is $h_1 = 4.5$, and $h_2 = 2.3$. Using these two kernel size and adjusted on consideration of data length of training set, we conduct another simulation about the identification error vs length of training data. We perform $L = 100$ repetitions for the average performance and the result is shown in Fig. 34.



(a) (b)

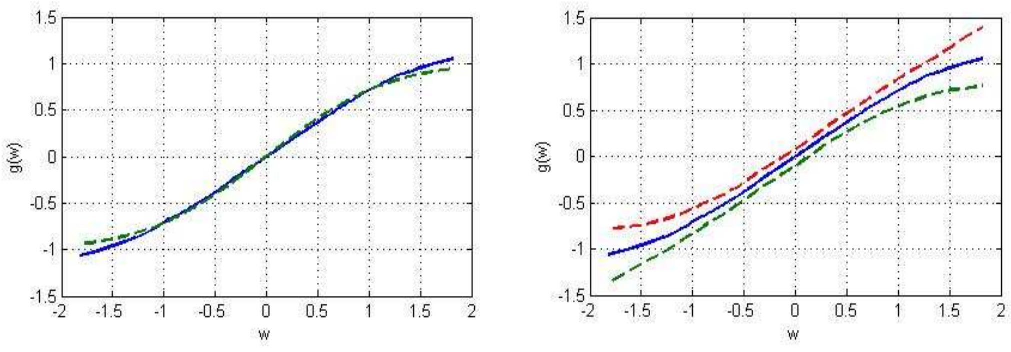
Figure 34: Semiparametric identification error vs length of training data. (a) Err($\hat{\theta}$) vs N, (b) Err($|\hat{g}(\hat{\gamma}^T\mathbf{U}) - m^*(\mathbf{U})|^2$) vs N, the solid line is the empirical error, while the dotted line is the regression fitting to the solid line. Note they almost coincide.

In Fig. 34(b), the error converges to some non-zero constant as $N \to \infty$. This is due to the error introduced by approximating the true MISO system using a semiparametric system. Thus we use the model Err($|\hat{g}(\hat{\gamma}^T\mathbf{U}) - m^*(\mathbf{U})|^2$) $= a \cdot N^b + c$ to fit the empirical observations in Fig. 34(b), and find the coefficients to be $a = 0.976$, $b = -0.788$ and $c = 0.01359$. The convergence rate -0.788 is very close to the theoretical value ($-2/5 \times 2$) for 1-dimensional nonparametric estimation (18), and the limit term 0.01359 is also very close to the calculations for the semiparametric projection error shown in Fig. 31.

For the case the length of training set $N = 150$, if we generate $L = 100$ independent training data sets and examine the nonparametric estimate of $\hat{g}(\cdot)$ in each simulation and compared them with the calculated optimal $g^*(w)$ when $\rho = 0$, the result is shown in Fig. 35.

3.5 Conclusions

In Section 3, semiparametric Hammerstein model identification schemes are examined by simulation studies. It shows that in many cases, MISO Hammerstein systems can be approximated by semiparametric models, such that the "Curse of Dimensionality" can be avoided. In semiparametric model identification, the estimation of parametric part "γ^*"



(a) (b)

Figure 35: Estimation of the nonparametric part in the semiparametric model. (a) Mean of $\hat{g}(\cdot)$ (solid line), and the true $g^*(\cdot)$ (dashed line), (b) Mean of $\hat{g}(\cdot)$ (solid line), and the 95% pointwise confidence interval of the estimate (dashed line).

and nonparametric part "$g^*(\cdot)$" are relatively two independent processes. In estimation of γ^*, minimization of MSE leads to better performance compared with derivative method. In estimating γ^* and estimating $g^*(\cdot)$, two kernel estimate bandwidths are involved (h_1 and h_2). The kernel bandwidths can be chosen by two possible approaches: One is to determine beforehand the optimal kernel size h_1 and h_2 individually and fix them in advance. The other approach is to determine h_1 by consider it as another variable, together with $\hat{\theta}$ (or $\hat{\gamma}$) should minimize the criterion function. Then use this selected $\hat{h}_1$ multiplied by a known constant factor to be h_2 the kernel bandwidth in estimation of $g^*(\cdot)$. Among the two approaches, the latter one will lead to more accurate estimation results, while the former one is faster to implement, and yet lead to estimation error relatively close to the latter approach. Re-sampling technique is also required in the process of estimating γ^*, and leave-one-out could be an optimal option when the training et is small. Partition method could also be applied when the training set is larger. The influence of nonlinear function shape, input dimensions, length of training data, correlation between different input dimensions, have also been demonstrated in the simulation studies.

4 Model Selection Algorithms for Identification of Hammerstein Systems

4.1 Introduction

This section of the thesis is concerned with identification of a SISO Hammerstein system depicted in Fig. 6. For a long time, researchers who have applied parametric approach to system identification have been relying on the assumption that the system nonlinearity $m^*(\cdot)$ can be specified by a finite dimensional parameter such that $m^*(\cdot) = m^*(\cdot;\theta), \theta \in \Theta$, i.e., the nonlinear function belongs to a certain known class of functions, e.g., a class of polynomial characteristics. A fundamental question, however, is the validity of the parametric assumption. This section will try to address this issue in the context of testing whether the nonlinear function $m^*(\cdot)$ in the Hammerstein model belongs to a certain parametric class.

In particular, we represent the system nonlinearity by an orthogonal expansion and our model selection methodology is based on a certain type of thresholding rules. The thresholding strategy is applied to estimated Fourier coefficients and it has its roots in the wavelet analysis of signals [25]. The thresholding algorithm is able to capture the sparse structure of the functional form as is it aimed to reduce the variance of the estimate at the expense of introducing some bias. Overall, the thresholding estimate of the system nonlinearity reveals the substantially smaller mean squared error. Yet another issue is discussed in Section 4: where the parametric assumption is not assumed, we can estimate the system nonlinearity by orthogonal expansion of a finite order. The required truncation parameter can be determined by certain shrinkage methods. Therefore, this approach is equivalent to estimate a general nonlinearity by a nested class of parametric functions.

Furthermore, we assume the input to the system to be i.i.d. Gaussian. Under this assumption, an orthogonal basis employing Hermite polynomials is a natural choice to represent the system nonlinearity. In fact, the Hermite polynomials form the orthogonal basis with respect to the Gaussian weight. This approach can also be extended to other input signal distributions, e.g., for uniformly distributed input one can use trigonometric functions or Legendre polynomials.

4.2 Orthogonal Series Estimation Using Hermite Polynomials

Hermite polynomials is a set of classical polynomials orthogonal with respect to a Gaussian weight, see [26], [12] for some basic properties of this polynomial. One of the common form of Hermite Polynomials is defined as follows:

$$H_n(x) = (-1)^n e^{x^2/2} \frac{d^n}{dx^n} e^{-x^2/2}.$$

It can be shown in [26] that the polynomials $\{H_n(x)\}$ satisfy the following orthogonality property

$$\int_{-\infty}^{\infty} H_k(x) H_l(x) w(x) dx = \begin{cases} k!, & \text{for } k = l \\ 0, & \text{for } k \neq l \end{cases}$$

where $w(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$. The first several Hermite polynomials are:

$$\begin{aligned}
H_0(x) &= 1 \\
H_1(x) &= x \\
H_2(x) &= x^2 - 1 \\
H_3(x) &= x^3 - 3x \\
H_4(x) &= x^4 - 6x^2 + 3 \\
H_5(x) &= x^5 - 10x^3 + 15x \\
H_6(x) &= x^6 - 15x^4 + 45x^2 - 15 \\
H_7(x) &= x^7 - 21x^5 + 105x^3 - 105x \\
H_8(x) &= x^8 - 28x^6 + 210x^4 - 420x^2 + 105
\end{aligned}$$

$$H_9(x) = x^9 - 36x^7 + 378x^5 - 1260x^3 + 945x$$
$$H_{10}(x) = x^{10} - 45x^8 + 630x^6 - 3150x^4 + 4725x^2 - 945.$$

The general means for generating Hermite polynomials is given by the following three-term recursive formula

$$H_{n+1}(x) = xH_n(x) - nH_{n-1}(x), \ \ n \geq 1,$$

with the initial conditions $H_0(x) = 1$, $H_1(x) = x$.

Let us define $h_k(x) = H_k(x)/\sqrt{k!}$. Then $\{h_k(x)\}$ constitutes an orthogonal basis, i.e., we have

$$\int_{-\infty}^{\infty} h_k(x)h_l(x)w(x)dx = \begin{cases} 1, & \text{for } k = l \\ 0, & \text{for } k \neq l. \end{cases}$$

Functions satisfying $\int |m(x)|^2 w(x)dx < \infty$ can be represented by Hermite polynomial expansions:

$$m(x) = \sum_{k=0}^{\infty} a_k h_k(x), \tag{41}$$

where

$$a_k = \int_{-\infty}^{\infty} m(x)h_k(x)w(x)dx \tag{42}$$

is the k-th Fourier coefficient. Due to the orthonomality we have Parseval's formula

$$\int_{-\infty}^{\infty} m^2(x)w(x)dx = \sum_{k=0}^{\infty} a_k^2. \tag{43}$$

The inner product associated with the Hermite expansion is defined with respect to the standard Gaussian density $w(\cdot)$, i.e., $< m_1, m_2 >= \int_{-\infty}^{\infty} m_1(x)m_2(x)w(x)dx$.

For the Hammerstein system in Fig. 6, i.e., $Y_n = \sum_{i=0}^{\infty} \lambda_i^* m^*(U_{n-i}) + \varepsilon_i$, with $\lambda_0^* = 1$, $\mathbb{E}m^*(U_n) = 0$, the relationship as in (12) allows us to expand the nonlinear characteristic $m^*(\cdot)$ in a series of Hermite polynomials in (41). Therefore $m^*(\cdot)$ can be estimated by the truncated version of the Hermite expansions. Hence, for a given data set $\{(U_1, Y_1), \cdots, (U_N, Y_N)\}$, we obtain

$$\hat{m}(u) = \sum_{k=0}^{T} \hat{a}_k h_k(u), \tag{44}$$

where

$$\hat{a}_k = \frac{1}{N} \sum_{n=1}^{N} Y_n h_k(U_n) \frac{w(U_n)}{f_U(U_n)} \tag{45}$$

is the estimate of a_k, T is the truncation parameter, and $f_U(\cdot)$ is the density function of the input signal. The formula in (45) can be derived in the following way:

$$\begin{aligned} a_k &= \int_{-\infty}^{\infty} m^*(u) h_k(u) w(u) du \\ &= \int_{-\infty}^{\infty} m^*(u) h_k(u) \frac{w(u)}{f_U(u)} f_U(u) du \\ &= \mathbb{E}\Big\{ m^*(U_n) h_k(U_n) \frac{w(U_n)}{f_U(U_n)} \Big\} \\ &= \mathbb{E}\Big\{ Y_n h_k(U_n) \frac{w(U_n)}{f_U(U_n)} \Big\}. \end{aligned} \tag{46}$$

The formula in (45) is just the empirical counterpart of (46). Particularly, if the input data are from the standard Gaussian distribution, (45) becomes

$$\hat{a}_k = \frac{1}{N} \sum_{n=1}^{N} Y_n h_k(U_n). \tag{47}$$

Note that if the input data have Gaussian distribution with non-standard variance, it can always be normalized by a linear transformation and then be treated as the normalized Gaussian distribution case. So the estimates in (44) and (47) can be applied to a more general class of cases where the input distribution is Gaussian with no unit variance.

The arguments used in (46) shows that $\hat{a}_k$ is the unbiased estimate of a_k, i.e.,

$$\mathbb{E}(\hat{a}_k) = a_k. \tag{48}$$

Note that this fact implied that $\mathbb{E}\hat{m}(u) = \sum_{k=0}^{T} a_k h_k(u)$ is the partial sum of representing $m^*(u)$ in terms of Hermite polynomials.

However, some more tedious algebra, see [12], reveals that

$$\text{var}(\hat{a}_k) = O(\frac{1}{N}). \tag{49}$$

It is important to note that (49) takes place regardless whether $a_k \neq 0$ or $a_k = 0$. The bounds in (49) holds under the assumption that the linear subsystem of the Hammerstein model is stable, i.e., $\sum_{i=0}^{\infty} |\lambda_i^*| < \infty$. Note also that the constant appearing in (49) depends on k. We can rewrite (49) in the following compact form

$$\hat{a}_k = a_k + O_P(N^{-1/2}), \qquad \text{for all } k \tag{50}$$

where $O_P(\cdot)$ denotes “in probability” convergence.

The meaning of (49) is that even if $a_k = 0$, then there exists irreducible statistical error contributing to the overall reconstructed accuracy of the nonlinear estimate $\hat{m}(u)$.

The aim of our research is to propose a modified estimate of a_k that is able to detect the case whether $a_k = 0$. Hence, we seek for an estimate $\tilde{a}_k$ with the following property

$$\tilde{a}_k = \begin{cases} a_k + O_P(N^{-1/2}), & \text{if } a_k \neq 0 \\ 0, & \text{if } a_k = 0. \end{cases}$$

For such an ideal estimator, one has to relax the property of being unbiased, i.e.,

$$\mathbb{E}(\tilde{a}_k) \neq a_k.$$

Nevertheless, the modified estimate $\tilde{a}_k$ can reveal the reduced mean squared error. A partial remedy to remove the undesirable property in (50) can be obtained by employing a class of thresholding rules, see Section 4.3.

4.3 Thresholding Techniques for Coefficients Shrinkage

In many cases, the coefficients $\{a_k; k = 0, 1, \cdots\}$ are sparse, meaning that most of them are zero except for a few values. Hence, denoting by $S_T = \{k : a_k \neq 0, 0 \leq k \leq T\}$ we have that the partial sum for representing $m^*(u)$ is given by $m_T(u) = \sum\limits_{k \in S_T} a_k h_k(u)$. The sparse case corresponds to the situation when $|S_T| \approx T^\varepsilon$ for some small $\varepsilon > 0$. In such circumstances the estimator $\hat{m}(u)$ in (44) is unable to detect a set of Fourier coefficients that are zero. As we have already noted in (50) this is due to the irreducible statistical error caused by the randomness of training data. As a result, the estimate $\hat{m}(u)$ is going to reveal a large reconstruction error. To address this important issue one needs to turn to some shrinkage approach often implemented in terms of thresholding rules where we can test the magnitude of Fourier coefficients and set them appropriately to zero if they are not in the sparse set of the underlying nonlinearity. Thresholding methods have been originally developed in the application of the statistical inference of wavelets [25], [27]. There are several types of thresholding rules, e.g., hard, soft, and block thresholding. Denote the estimated coefficients after thresholding by $\tilde{a}_k$. Hard thresholding corresponds to

$$\tilde{a}_k = \begin{cases} 0, & |\hat{a}_k| < th, \\ \hat{a}_k, & |\hat{a}_k| \geq th. \end{cases} \tag{51}$$

where th is the threshold to be selected. One simple way of selecting th is the so-called "Universal Threshold" [28], in which th is specified as $\sqrt{\widehat{\text{var}(\hat{a}_k)} \cdot 2 \log N}$, and $\widehat{\text{var}(\hat{a}_k)}$ is an estimate of the variance of $\hat{a}_k$, see Section 4.7. Thresholding Hermite coefficients could be particularly useful if we know that the true characteristic is a polynomial of some unknown order.

It is worth mentioning that $\tilde{a}_k$ is a biased estimate of a_k. This is a general property of the shrinkage rules, i.e., unbiasedness must be relaxed in order to capture a sparse structure of the underlying characteristic.

4.4 Estimating a General Nonlinearity by Hermite Polynomials

The estimate $\hat{m}(\cdot)$ in (44) is able to converge under some conditions on $m^*(\cdot)$ to a general class of nonlinear characteristics $m^*(\cdot)$, provided that the truncation parameter T depends on the training size N in such a way that

$$T(N) \to \infty$$

and

$$\frac{\sqrt{T(N)}}{N} \to 0$$

as $N \to \infty$. The proofs of such asymptotic properties of the Hermite series nonparametric estimate $\hat{m}(\cdot)$ can be found in [12].

In this research, however, we are interested in a finite sample size property of $\hat{m}(\cdot)$, i.e., the problem of choosing the truncation parameter T. Our strategy for specifying T is relying on the global discrepancy measure between $\hat{m}(u)$ and $m^*(u)$. Owing to Paserval's formula we first represent the Integrated Square Error (ISE) as follows

$$\begin{aligned} \text{ISE}(\hat{m}) &= \int_{-\infty}^{\infty} (\hat{m}(u) - m^*(u))^2 w(u) du \\ &= \sum_{k=0}^{T} (\hat{a}_k - a_k)^2 + \sum_{k=T+1}^{\infty} a_k^2. \end{aligned} \tag{52}$$

Next, the Mean Integrated Square Error (MISE) can be evaluated by

$$\begin{aligned} \text{MISE}(\hat{m}) &= \mathbb{E}\Big\{\text{ISE}(\hat{m})\Big\} \\ &= \sum_{k=0}^{T} \mathbb{E}(\hat{a}_k - a_k)^2 + \sum_{k=T+1}^{\infty} a_k^2 \\ &= \sum_{k=0}^{T} \text{var}(\hat{a}_k) + \sum_{k=T+1}^{\infty} a_k^2, \end{aligned} \tag{53}$$

where the relationship in (48) was used and the expected sign is taken with respect to all training sets of size N.

The minimizer of (53) with respect to T determines the optimal truncation value T^* of the orthogonal estimate, i.e.,

$$\begin{aligned} T^* &= \arg\min_{T\geq 1} \text{MISE}(\hat{m}) \\ &= \arg\min_{T\geq 1} \Big\{ \sum_{k=0}^{T} \text{var}(\hat{a}_k) + \sum_{k=T+1}^{\infty} a_k^2 \Big\}. \end{aligned} \tag{54}$$

This defines the theoretical optimal (with respect to MISE) value of the truncation parameter that can not be directly computed as it contains unknown variables. In fact, both var$(\hat{a}_k)$ and a_k^2 are unknown in advance. A computable estimate of T^* can be obtained by estimating var$(\hat{a}_k)$ and a_k^2. Let $\widehat{\text{var}(\hat{a}_k)}$ be some estimate of var$(\hat{a}_k)$, see Section 4.7. Since $0 \leq a_k^2 = \mathbb{E}(\hat{a}_k^2) - \text{var}(\hat{a}_k)$ then one can estimate a_k^2 by $\Big(\hat{a}_k^2 - \widehat{\text{var}(\hat{a}_k)}\Big)$. To prevent that this difference to be negative, we estimate a_k^2 by

$$\Big(\hat{a}_k^2 - \widehat{\text{var}(\hat{a}_k)}\Big)_+,$$

where $a_+ = \max(a, 0)$.

All these considerations yield the following empirical choice of the truncation parameter

$$\begin{aligned} \hat{T} &= \arg\min_{1\leq T\leq N} \Big\{ \sum_{k=0}^{T} \widehat{\text{var}(\hat{a}_k)} + \sum_{k=T+1}^{N} (\hat{a}_k^2 - \widehat{\text{var}(\hat{a}_k)})_+ \Big\} \\ &= \arg\min_{1\leq T\leq N} \Big\{ \sum_{k=0}^{T} \widehat{\text{var}(\hat{a}_k)} - \sum_{k=0}^{T} (\hat{a}_k^2 - \widehat{\text{var}(\hat{a}_k)})_+ \Big\} \end{aligned} \tag{55}$$

where $\hat{T}$ is believed to be the estimate of T^*. The final formula in (55) results from the fact that $\sum_{k=T+1}^{\infty} a_k^2 = \int_{-\infty}^{\infty} m^{*2}(u)w(u)du - \sum_{k=0}^{T} a_k^2$.

We can conjecture that $\hat{T} \to T^*$ (P) as $N \to \infty$. Nevertheless, such asymptotic properties of $\hat{T}$ are postponed to future research. In this thesis, the selection of truncation parameter $\hat{T}$ is examined by means of simulation studies.

4.5 Testing for the Polynomial Form of the Nonlinear Characteristic

While Section 4.3 have discussed the issue of Hermite polynomial estimation and coefficient shrinkage for estimating the polynomial characteristics, it is of interest whether it is reasonable to assume the nonlinear function to be polynomial. The problem here would be testing the polynomial model hypothesis against all other nonparametric alternitives, i.e:

$$H_0: \quad m^*(\cdot) \text{ is a polynomial of order less than } p, \tag{56}$$

against $\qquad H_a: \quad m^*(\cdot) \text{ is not a polynomial of order less than } p.$

This is equivalent to testing

$$H_0: \quad r^*(u;p) \text{ is a constant for each } u \in \mathbb{R}, \tag{57}$$

where $r^*(u;p) = m^*(u) - \sum_{k=0}^{p-1} a_k^* h_k(u)$, and $\{a_k^*,\ k = 0, \cdots, p\}$ are the Hermite coefficients for $m^*(\cdot)$, against the alternative:

$$H_a: \quad r^*(\cdot;p) \text{ is not a constant for each } u \in \mathbb{R}.$$

Note this test is actually the no-effect hypothesis for $r^*(\cdot;p)$.

The expression $r^*(\cdot;p)$ can be estimated by:

$$\hat{r}(u;p) = \sum_{k=p}^{p+T'} \hat{a}_k h_k(u), \tag{58}$$

where T' is the truncation parameter and $\hat{a}_k$ is the same as that appears in (44) and (47). In follows from the central limit theorem, $\hat{a}_k \sim \mathbf{N}(a_k, \sqrt{\text{var}(\hat{a}_k)})$. Under the null hypothesis, $\hat{a}_k/\sqrt{\widehat{\text{var}(\hat{a}_k)}}$, $k = p, \cdots$ would have standard normal distribution for asymptotically large N. This leads to the Neyman Smooth Test [29], [30], [31] statistic:

$$S_N = \sum_{k=p}^{p-1+b} \frac{\hat{a}_k^2}{\widehat{\text{var}(\hat{a}_k)}}. \tag{59}$$

It is postulated that $\hat{a}_k/\sqrt{\widehat{\text{var}(\hat{a}_k)}}$, $k = p, \cdots$ are independent of each other. Under the null hypothesis, T_N should have χ^2 distribution with degrees of freedom b. For a statistical test of significance level α, the one-sided χ^2-test would reject the null hypothesis H_0 for $S_N > \chi^2(\alpha; b)$, where $\chi^2(\alpha; b)$ denotes the $1-\alpha$ quantile of the cdf of χ^2_b.

Neyman suggested that the selection of b could be important for the performance of this test, for a too large value of b would undermine the influence of the abnormal terms of $\hat{a}_k/\sqrt{\widehat{\text{var}(\hat{a}_k)}}$. He proposed b should be set as a priori. In the following simulation studies, we fix this value to be $b = 10$.

In the simulation studies, it is shown that Neyman test can really distinguish between a true polynomial structure Hammerstein system and a non-polynomial structure of Hammerstein system. However, a larger data set is required to test $m^*(\cdot)$ to qualify higher order polynomial assumptions. Besides, $\hat{a}_k$ as well as $\widehat{\text{var}(\hat{a}_k)}$ should be carefully estimated. Weighted Monte Carlo (Section 4.6) would be a way to improve the estimate in (47). The estimation of $\text{var}(\hat{a}_k)$ will be discussed in Section 4.7. More detailed examples will be shown in the simulation studies in Section 4.8.

4.6 Weighted Monte Carlo

The derivation in (46) can be considered as a Monte Carlo problem. As such the formula in (47) is the simplest Monte Carlo technique for the integral evaluation. We have already pointed out in (49) that the variance of the estimate $\hat{a}_k$ decreases as $O(\frac{1}{N})$. Our goal is to apply refined Monte Carlo integration algorithms developed in [32] in order to improve the $\frac{1}{N}$ convergence rate. To do so, let $\{(U_{(1)}, Y_{[1]}), \cdots, (U_{(n)}, Y_{[n]}), \cdots, (U_{(N)}, Y_{[N]})\}$ be the ordered version of $\{(U_1, Y_1), \cdots, (U_N, Y_N)\}$ with respect to U_n. Then (47) can be rewritten as

$$\hat{a}_k = \sum_{n=1}^{N} \frac{1}{N} Y_{[n]} h_k(U_{(n)}),$$

which resembles the rectangular method of numerical integration. Yet, an improvement is to replace the rectangular-form of numerical integration by trapezoidal rule, in particular, the weighted Monte Carlo method proposed by Yakowitz et al [32]:

$$\hat{a}_k = \sum_{n=1}^{N} \omega_n Y_{[n]} h_k(U_{(n)}), \tag{60}$$

where $\omega_n = \frac{1}{2}\Big(\Phi^{-1}(U_{(n+1)}) - \Phi^{-1}(U_{(n-1)})\Big)$, $\Phi^{-1}(\cdot)$ is the inverse standard Gaussian cdf, and $U_{(0)} = -\infty$, $U_{(N+1)} = +\infty$. It can be shown that the estimator in (60) has a much faster convergence rate:

$$\mathbb{E}\{(\hat{a}_k - a_k)^2\} = o(1/N^4).$$

The implementation of such weighted Monte Carlo method will also be examined in simulation studies.

4.7 Estimation of the Variance

In both Section 4.3 and Section 4.4, the variance of the estimator $\text{var}(\hat{a}_k)$ needs to be estimated. The simplest way is to estimate it directly through the observed data. Based on (47):

$$\text{var}(\hat{a}_k) = \text{var}\Big(\frac{1}{N}\sum_{n=1}^{N} Y_n h_k(U_n)\Big)$$

Using Lemma 12.4 in [12], we can prove that

$$\text{var}(\hat{a}_k) = \frac{1}{N}\text{var}\Big(Y_1 h_k(U_1)\Big), \qquad \text{for } \ k \geq 1. \tag{61}$$

Let us estimate $\text{var}(\hat{a}_k)$ by a standard formula

$$\widehat{\text{var}(\hat{a}_k)} = \frac{1}{N}\frac{1}{N-1}\sum_{n=1}^{N}\Big(Y_n h_k(U_n) - \hat{a}_k\Big)^2. \tag{62}$$

It is worth mentioning that although the estimator (62) is consistent, yet it is not unbiased, due to the dependency of Y_is.

Another approach to estimate $\widehat{\text{var}(\hat{a}_k)}$ is bootstrapping. Efron et al [33] shows that generating the bootstrapping sets according to the pair (U_n, Y_n) is a relatively good approach for estimation. The realization of such bootstrapping for estimating $\widehat{\text{var}(\hat{a}_k)}$ can be generalized in the following procedures:

1. Given the observations $\{(U_1, Y_1), \cdots, (U_N, Y_N)\}$, draw $(U_n^{[boot]}, Y_n^{[boot]})$, $n = 1, \cdots, N$ from it with replacement, this forms a bootstrapping set $\{(U_1^{[boot]}, Y_1^{[boot]}), \cdots, (U_N^{[boot]}, Y_N^{[boot]})\}$.

2. Compute $\hat{a}_k^{[boot]} = \sum_{n=1}^{N} Y_n^{[boot]} h_k(U_n^{[boot]})$, $k = 0, 1, \cdots$.

3. Repeat steps 1 and 2, B times to get $\hat{a}_k^{[boot,1]}, \cdots, \hat{a}_k^{[boot,B]}$.

4. Estimate var$(\hat{a}_k)$ by:

$$\widehat{\text{var}(\hat{a}_k)} = \frac{1}{N}\frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{a}_k^{[boot,b]} - \frac{1}{B}\sum_{t=1}^{B}\left(\hat{a}_k^{[boot,b]}\right)\right)^2, \quad k = 0, 1, \cdots \tag{63}$$

For the estimate $\hat{a}_k$ in (47), both (62) and (63) can be used to calculate $\widehat{\text{var}(\hat{a}_k)}$. It is shown in Section 4.8.1 that estimate $\widehat{\text{var}(\hat{a}_k)}$ according to (63) and that according to (62) leads to similar performance in thresholding the coefficients. So the formula in (62) is preferred since bootstrapping is usually much more time consuming. However, if $\hat{a}_k$ is obtained by the modified Monte Carlo (60), then (62) can no longer be applied to estimate the variance of $\hat{a}_k$, $\widehat{\text{var}(\hat{a}_k)}$ must be calculated through bootstrapping method (63).

4.8 Simulation Studies

4.8.1 Hermite Polynomial Estimation with Thresholding Rule

Experiment 1 Suppose we observe a data set from the following system:

- Input $U \sim N(0, 1)$.

- Linear part of the system is characterized by FIR(3), $\Lambda = [1, -0.8, 0.6, -0.4]$, whereas the same identification method also be applied to IIR subsystems (e.g. ARMA models).

- $m^*(u) = h_1(u) + h_2(u) + h_3(u) \simeq 0.482u^3 + 0.707u^2 - 0.225u - 0.707$.
- $\varepsilon_n \sim$ i.i.d. $N(0, 0.648)$. The amplitude of noise here corresponds to a 10 dB signal-to-noise ratio in the system.
- Length of data observed: $N = 5000$.
- Obtain Hermite coefficients $\hat{a}_k$ by (47). Use bootstrapping to estimate $\widehat{\text{var}(\hat{a}_k)}$, and generate $B = 100$ bootstrapping sets. Then apply thresholding rule and then obtain the modified coefficients $\tilde{a}_k$. Repeat this for $L = 100$ times.

The simulation result is shown in Fig. 36.

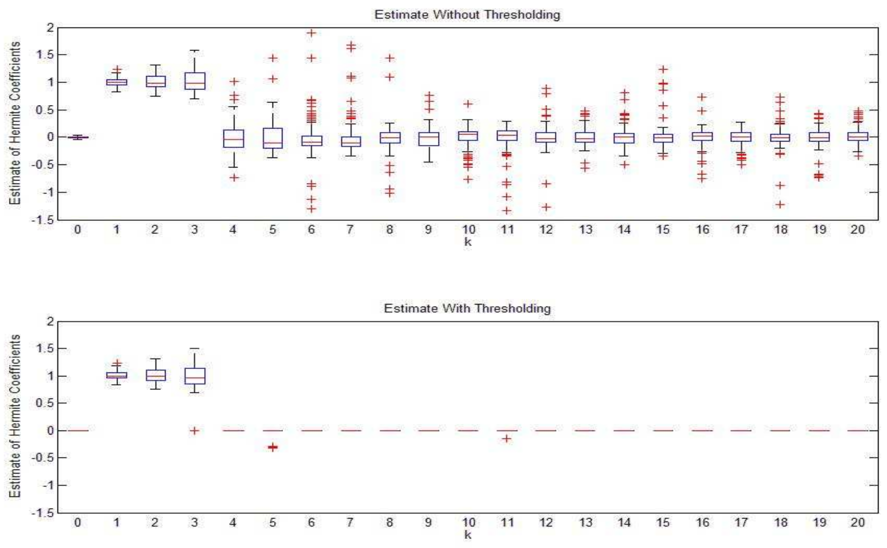


Figure 36: Comparison of the boxplot of Hermite polynomial estimate without and with thresholding rules in Experiment 1, Section 4.8.1. Note that the true value should be $\alpha_k^* = 1$ for $k = 1, 2, 3$ and $\alpha_k^* = 0$ for other value of k. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers are plotted individually.

Fig. 36 shows that applying orthogonal estimate crudely will always result in nonzero coefficients even though many of them actually should be zero. By contrast, thresholding method can eliminate most of the redundant coefficients. If we calculate the MISE of the estimate, then we find shrinkage method will greatly enhance the estimate (MISE 0.1352 vs MISE 4.7907).

In this experiment, if we use (62) rather than (63) to calculate $\widehat{\text{var}(\hat{a}_k)}$, their difference will be shown in Fig. 37. There is very little difference in most of the cases. If we use the shrinked coefficient estimates $\tilde{a}_k$ to form the nonparametric estimate $\hat{m}(\cdot)$ and then to calculate the MISE of the Estimate, then the above two approaches would lead MISE to be 0.1662 and 0.1576, which again shows their performances comparably similar.

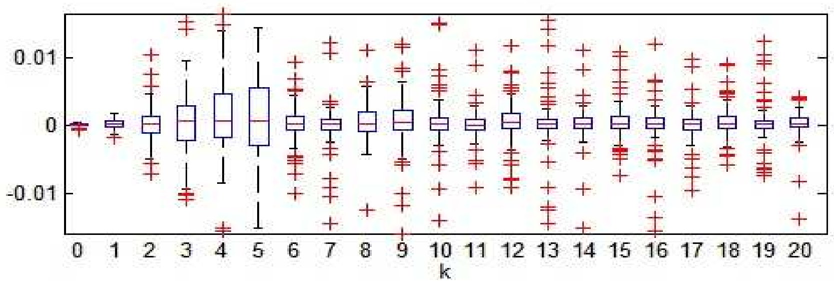


Figure 37: Boxplot of the difference of calculating $\widehat{\text{var}(\hat{a}_k)}$ by (62) and by (63).

Experiment 2 Suppose the nonlinear system with 4th order polynomial, then with all other conditions similar to the above experiment:

- Input $U \sim N(0, 1)$.
- Linear subsystem: $\Lambda = [1, -0.8, 0.6, -0.4]$.
- $m^*(u) = h_1(u) + h_2(u) + h_3(u) + h_4(u) \simeq 0.204u^4 + 0.482u^3 - 0.518u^2 - 0.225u - 0.095$.
- $\varepsilon_n \sim$ i.i.d. $N(0, 0.864)$. The amplitude of noise here corresponds to a 10 dB signal-to-noise ratio in the system.
- Length of data observed: $N = 5000$.
- Obtain Hermite coefficients $\hat{a}_k$ by (47) and apply thresholding rule and then obtain the modified coefficients $\tilde{a}_k$. Repeat this for $L = 100$ times.

The result is shown in Fig. 38.

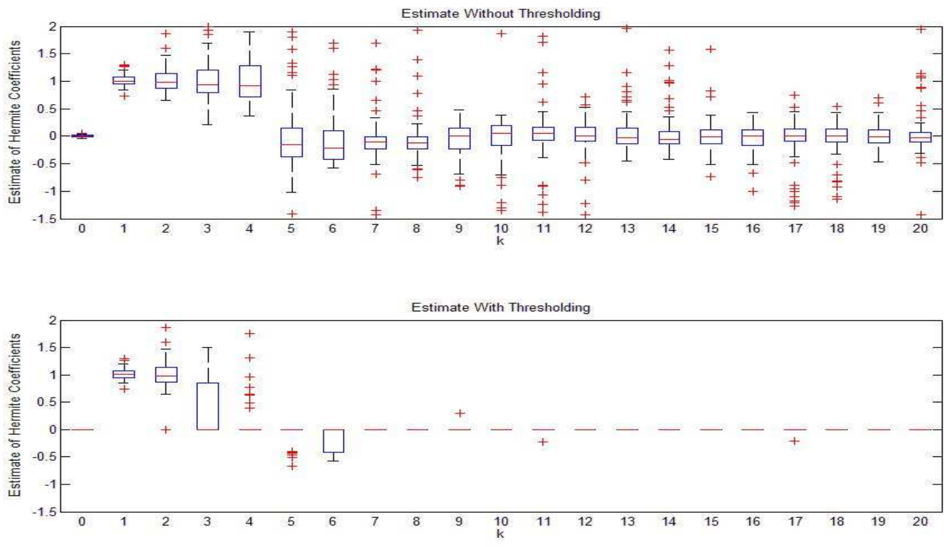


Figure 38: Comparison of the boxplot of Hermite polynomial estimate without and with thresholding rules in Experiment 2, Section 4.8.1. Note that the true value should be $\alpha_k^* = 1$ for $k = 1, 2, 3, 4$ and $\alpha_k^* = 0$ for other value of k.

Unlike estimating the system with polynomial of order 3 in the previous experiment, the estimate in Fig. 38 behaves badly for $\tilde{a}_3$ and $\tilde{a}_4$. If we increase the length of data set to a larger value and conduct a similar simulation, this problem could be solved. It will be shown in the next experiment.

Experiment 3

- Input $U \sim N(0, 1)$.
- Linear subsystem: $\Lambda = [1, -0.8, 0.6, -0.4]$.
- $m^*(u) = h_1(u) + h_2(u) + h_3(u) + h_4(u) \simeq 0.204u^4 + 0.482u^3 - 0.518u^2 - 0.225u - 0.095$.
- $\varepsilon_n \sim$ i.i.d. $N(0, 0.864)$. The amplitude of noise here corresponds to a 10 dB signal-to-noise ratio in the system.

- Length of data observed: $N = 20000$.
- Obtain Hermite coefficients $\hat{a}_k$ by (47) and apply thresholding rule and then obtain the modified coefficients $\tilde{a}_k$. Repeat this for $L = 100$ times.

The result is shown in Fig. 39. Comparing with the previous simulation, it suggests that accurately determining higher order of polynomial characteristics requires a bigger data set observed. Some simple algebra shows the for the same training data, the term var$(\hat{a}_k)$ grows substantially as k increases, therefore making the "Universal threshold" in (51) increase substantially for larger value of k. This will lead to $\tilde{a}_k$ being set to zero for larger value of k regardless of the true nonlinear characteristic. Only by using larger training data set could solve this problem. Thus thresholding method in Hermite series estimate works for smaller orders of polynomial, and would require larger data set to work for larger orders.

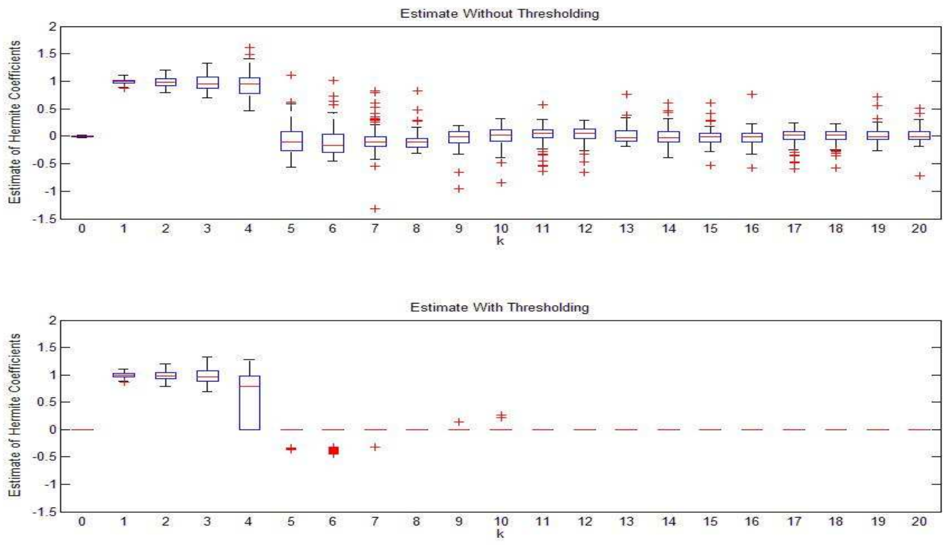


Figure 39: Comparison of the boxplot of Hermite polynomial estimate without and with thresholding rules in Experiment 3, Section 4.8.1. Note that the true value should be $\alpha_k^* = 1$ for $k = 1, 2, 3, 4$ and $\alpha_k^* = 0$ for other value of k.

4.8.2 Approximating the Nonlinear Characteristics by Polynomial Function of Finite Orders

Experiment 1

- Input $U \sim N(0, 1)$.
- Linear subsystem: $\Lambda = [1, -0.8, 0.6, -0.4]$.
- $m^*(\cdot) = \arctan(\cdot)$.
- $\varepsilon_n \sim$ i.i.d. $N(0, 0.097)$. The amplitude of noise here corresponds to a 10 dB signal-to-noise ratio in the system.
- Length of data observed: $N = 1000$.
- Obtain Hermite coefficients $\hat{a}_k$ by (47). Compare the following ways:
  1. Apply thresholding rule and then obtain the modified coefficients $\tilde{a}_k$, consider the highest order of coefficients $\tilde{a}_k$ that is not zero to be the order of the polynomial approximation, denote this order by $\hat{T}$.
  2. Use $\hat{T}$ in (55) as truncation parameter and use $\{\hat{a}_k;\ k = 0, \cdots, \hat{T}\}$ as the final coefficients in the estimate.
- Use a testing set of 10000 data to approximate the Integrated Square Error (ISE) of the estimate.
- Repeat the above estimation for $L = 100$ times.

If $\hat{a}_k$ is calculated by (47), the result is shown in Table 3. The approximation of $m^*(\cdot) = \arctan(\cdot)$ by Hermite polynomial estimate is shown in Fig. 40.

If we use the weighted Monte Carlo integration (60) to calculate $\hat{a}_k$ in the previous simulation, the exact same data sets lead to the following estimation result shown in Table 4.

	Use Thresholding	Obtained from (55)
Median of $\hat{T}$	1	4
25% quantile of $\hat{T}$	1	3
75% quantile of $\hat{T}$	3	8
Mean of ISE	0.01602904602	0.01687971919
SD of ISE	0.00748799189	0.01596121436

Table 3: Using Hermite polynomial estimate to approximate a non-polynomial function in the nonlinear part in a Hammerstein system, with $\hat{a}_k$ calculated by (47), and length of training data equal to 1000.

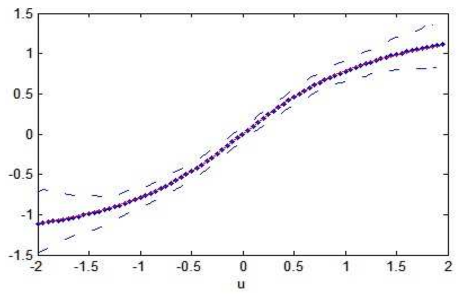


Figure 40: The approximation of $m^*(\cdot) = \arctan(\cdot)$ by finite order Hermite polynomial expansion. Use $\hat{T}$ in (55) as the truncation parameter. The solid line represents the true $m^*(\cdot)$, while the dot represent the mean of the estimate for different sets of data at the sampled points, which almost concide with the solid line. The dashed line represents for 95% pointwise confidence interval of the estimate.

	Use Thresholding	Obtained from (55)
Median of $\hat{T}$	3	13
25% quantile of $\hat{T}$	3	5
75% quantile of $\hat{T}$	3	38
Mean of ISE	0.00920190510	0.01766346591
SD of ISE	0.00564368652	0.02112562867

Table 4: Using Hermite polynomial estimate to approximate a non-polynomial function in the nonlinear part in a Hammerstein system, with $\hat{a}_k$ calculated by weighted Monte Carlo (60), and length of training data equal to 1000..

Experiment 2 If we use a longer data set, and conduct a similar experiment as the previous one:

- Input $U \sim N(0, 1)$.
- Linear subsystem: $\Lambda = [1, -0.8, 0.6, -0.4]$.

- $m^*(\cdot) = \arctan(\cdot)$.
- $\varepsilon_n \sim$ i.i.d. $N(0, 0.097)$. The amplitude of noise here corresponds to a 10 dB signal-to-noise ratio of the system.
- Length of data observed: $N = 5000$.
- Obtain Hermite coefficients $\hat{a}_k$. Compare the following ways:
  1. Apply thresholding rule and then obtain the modified coefficients $\tilde{a}_k$, consider the highest order of coefficients $\tilde{a}_k$ that is not zero to be the order of the polynomial approximation.
  2. Use $\hat{T}$ in (55) as truncation parameter and use $\{\hat{a}_k;\ k = 0, \cdots, \hat{T}\}$ as the final coefficients in the estimate.
- Use a testing set of 10000 data to approximate the Integrated Error (ISE) of the estimate.
- Repeat the above estimation for $L = 100$ times.

If $\hat{a}_k$ is calculated by (47), the result is shown in Table 5.

	Use Thresholding	Obtained from (55)
Median of $\hat{T}$	3	7
25% quantile of $\hat{T}$	3	7
75% quantile of $\hat{T}$	3	11
Mean of ISE	0.0047361547	0.00358710897
SD of ISE	0.00333382545	0.00246170059

Table 5: Using Hermite polynomial estimate to approximate a non-polynomial function in the nonlinear part in a Hammerstein system, with $\hat{a}_k$ calculated by (47), and length of training data equal to 5000.

The approximation of $m^*(\cdot) = \arctan(\cdot)$ by Hermite polynomial estimate is shown in Fig. 41:

If we use the weighted Monte Carlo integration in the previous simulation, the exact same data sets lead to the following estimation result shown in Table 6.

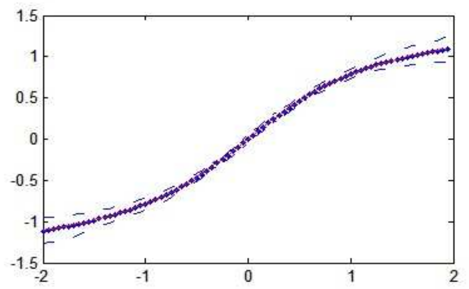


Figure 41: The approximation of $m^*(\cdot) = \arctan(\cdot)$ by finite Hermite polynomial estimate. Use $\hat{T}$ in (55) as the truncation parameter. The solid line represents the true $m^*(\cdot)$, while the dots represent the mean of the estimate for different sets of data at the sampled points, which almost coincide with the solid line. The dashed line represents for 95% pointwise confidence interval of the estimate.

	Use Thresholding	Minimize SURE
Median of $\hat{T}$	5	11
25% quantile of $\hat{T}$	3	9
75% quantile of $\hat{T}$	7	19
Mean of ISE	0.00291291034	0.00236654928
SD of ISE	0.00164455304	0.00131312311

Table 6: Using Hermite polynomial estimate to approximate a non-polynomial function in the nonlinear part in a Hammerstein system, with $\hat{a}_k$ calculated by (60), and length of training data equal to 5000.

Comparing this result with the previous experiment, it clearly demonstrates that for the same system and same input, the optimal truncation parameter increases as the length of data set increases.

4.8.3 Testing the Polynomial Hypothesis of the Nonlinear Characteristics

In this section, simulations are conducted to test whether the nonlinear function in a Hammerstein model belongs to a polynomial parametric class. Hammerstein system with true polynomial nonlinear structures and non-polynomial (Inverse tangent) nonlinear structures are compared. A FIR(3) linear part is assumed to be known, and a medium level noise is present, while the SNR for the system is 10dB.

Experiment 1 Suppose we observe data sets from the following two systems with non-linear function $m_1^*(\cdot)$ and $m_2^*(\cdot)$:

- Input $U \sim N(0,1)$.
- Linear part of the system is characterized by FIR(3): $\Lambda = [1, -0.8, 0.6, -0.4]$, whereas the same identification method also applies to IIR subsystems (e.g. ARMA models).
- $m_1^*(u) = h_1(u) + h_2(u) + h_3(u) \simeq 0.482u^3 + 0.707u^2 - 0.225u - 0.707$.
- $m_2^*(u) = \arctan(u)$.
- $\varepsilon_{1,n} \sim$ i.i.d. $N(0, 0.648)$, $\varepsilon_{2,n} \sim$ i.i.d. $N(0, 0.097)$. The amplitude of noise here corresponds to a 10 dB signal-to-noise ratio in each of the two systems respectively.
- Length of data observed: $N = 5000$.
- Obtain Hermite coefficients $\hat{a}_k$. Use (62) to estimate $\widehat{\text{var}(\hat{a}_k)}$.
- Test the hypothesis H_0: $m_j^*(\cdot)$ is polynomial of order less than p. $j = 1, 2$, $p = 1, 2, \cdots, 20$. Use Neyman Test described in Section 4.5 with $b = 10$. Obtain the P-Value of the test corresponding to testing of each data set.
- Repeat this for $L = 100$ times. Show the boxplot of P-Value corresponding to different value of p in each data set among the L repetitions. Also obtain the empirical probability of rejecting H_0 at significance level $\alpha = 0.05$ based on these L trials.

With $\hat{a}_k$ calculated by (47), the result is shown in Fig. 42.
If the $\hat{a}_k$ is calculated by weighted Monte Carlo (60), the result is shown in Fig. 43.

It is not surprising to observe testing "$m_1^*(\cdot)$ is polynomial and its order is less than $p = 1$ (or 2, 3)" is always rejected. For $p \geq 4$, the tests for $m_1^*(\cdot)$ usually have large P-Value, which means not rejecting H_0. Compare Fig. 42 with Fig. 43, definitely using weighted Monte Carlo would increase the accuracy of testing, since the probability that not

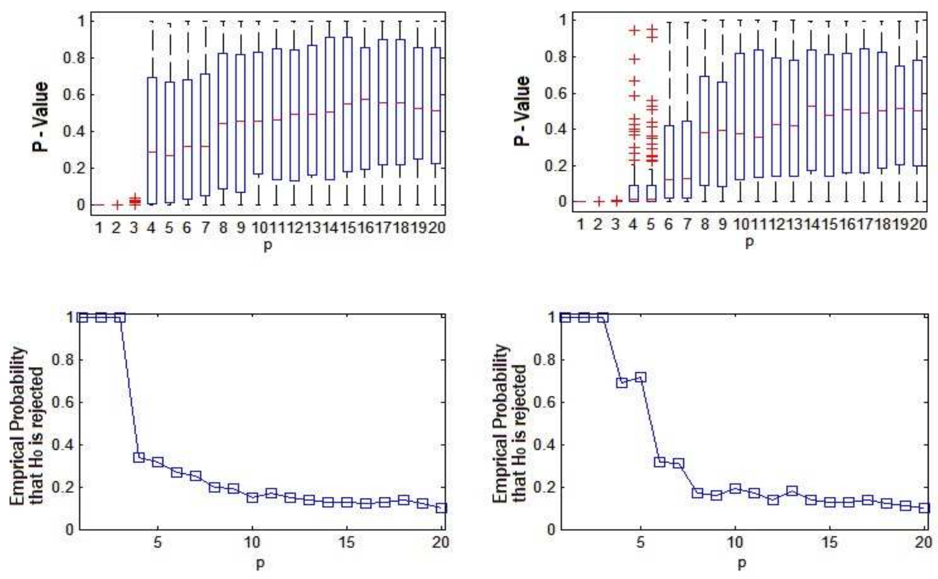


Figure 42: Plot of Neyman test result when $\hat{a}_k$ is obtained by (47) in Example 1, Section 4.8.3. The two columns correspond to testing the two systems $m_1^*(\cdot)$ (true polynomial) and $m_2^*(\cdot)$ (inverse tangent). The two rows show the boxplot of the P-Value corresponds to testing hypothesis of different orders, as well as the empirical probability of rejecting null hypothesis based on the L trials.

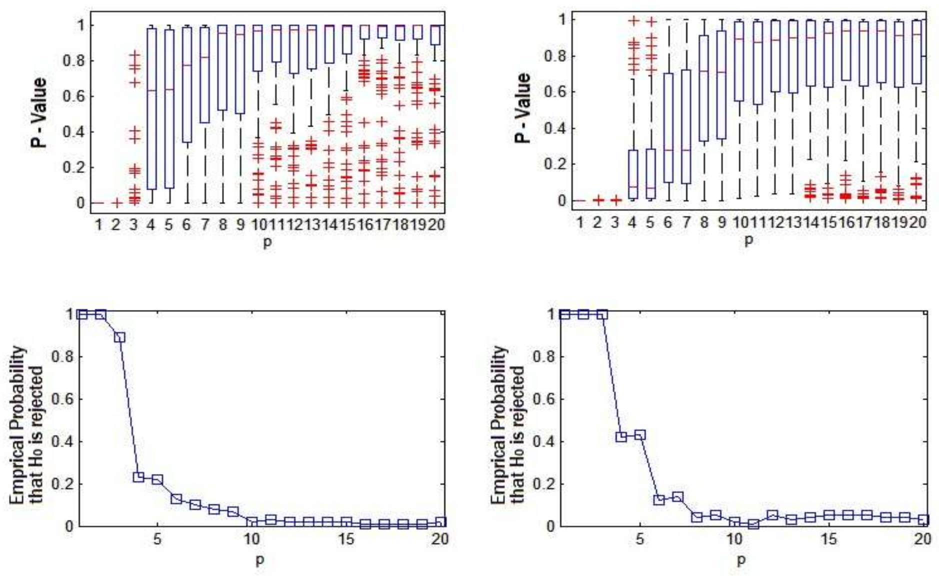


Figure 43: Plot of hypothesis testing result when $\hat{a}_k$ is estimated by (60) in Example 1, Section 4.8.3. The two columns correspond to testing the two systems $m_1^*(\cdot)$ (true polynomial) and $m_2^*(\cdot)$ (inverse tangent). The two rows show the boxplot of the P-Value corresponds to testing hypothesis of different orders, as well as the empirical probability of rejecting null hypothesis based on the L trials.

rejecting H_0 for $p \geq 4$ in testing $m_1^*(\cdot)$ is larger. Thus Neyman test usually shows $m_1^*(\cdot)$ can be viewed as a polynomial of order 3.

It is noticeable that it failed to correctly test the system with $m_2^*(\cdot)$ nonlinearity for $p \geq 4$. The plot shows that the test could not reject the hypothesis that $m_2^*(\cdot)$ is polynomial of order less than p when $p \geq 4$. This is due to the fact that the true Hermite polynomial coefficients $a_{2;k}^* =$ for $m_2^*(\cdot)$ is small for higher orders, i.e: $a_{2;1}^* = 0.65568$, $a_{2;3}^* = -0.127112$, $a_{2;5}^* = 0.0508282$, $a_{2;7}^* = -0.0258004$, $a_{2;9}^* = 0.0148394$, $a_{2;11}^* = -0.00923285$, $a_{2;13}^* = 0.00606755$, $a_{2;15}^* = -0.00415276$, ..., and $a_{2;k}^* = 0$, for $k = 0, 2, 4 \cdots$. Identifying the nonzero existence of higher order coefficients through testing would be possible only if var$(\hat{a}_k)$ is small for all k. In the next experiment a larger data set is used, and it shows the validity of testing higher order polynomial orders.

Experiment 2 The experiment is the same from last one except a much bigger size of data set is used:

- $N = 100000$.

Using $\hat{a}_k$ calculated by (47) and by (60) will lead to the result in Fig. 44 and Fig. 45 respectively.

With a large training set, testing larger order hypothesis of $m_2^*(\cdot)$ becomes possible. "$m_2^*(\cdot)$ is polynomial and its order is less than p" can be always rejected for p up to 9 or 10. It can be concluded that with large enough data, testing the polynomial order hypothesis of any nonlinear system within a Hammerstein system can be examined in the same way.

Besides, comparing Fig. 42 with Fig. 43, and comparing Fig. 44 with Fig. 45, it shows using weighted Monte Carlo (60) to estimate $\hat{a}_k$ will lead to a higher correct rate in testing $m_1^*(\cdot)$.

4.9 Conclusions and Future Work

In Section 4, Hermite polynomial series estimate in SISO Hammerstein system identification is examined. Simulation shows with large training data set, the parametric assumption

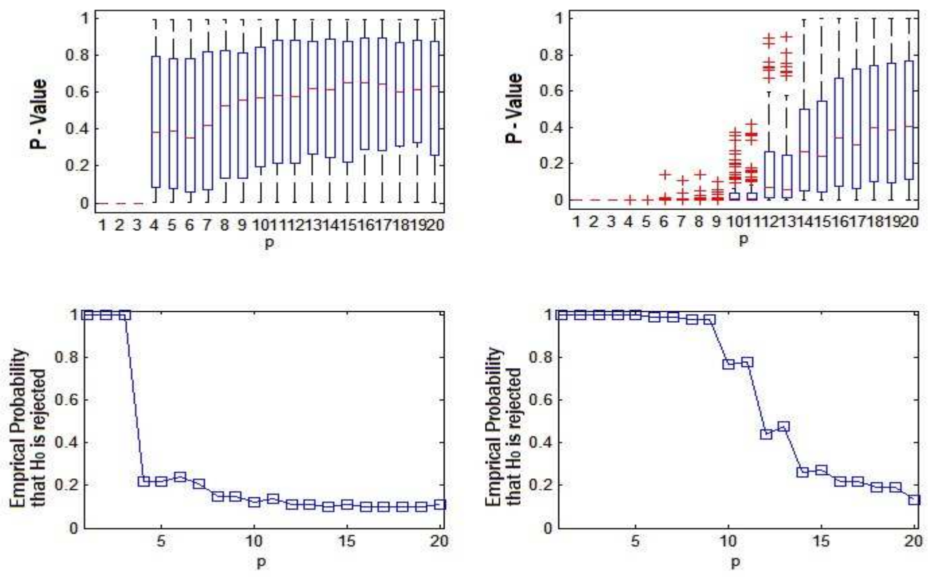


Figure 44: Plot of Neyman test result with $\hat{a}_k$ calculated by (47) in Example 2, Section 4.8.3. The two columns correspond to testing the two systems $m_1^*(\cdot)$ (true polynomial) and $m_2^*(\cdot)$ (inverse tangent). The two rows show the boxplot of the P-Value corresponds to testing hypothesis of different orders, as well as the empirical probability of rejecting null hypothesis based on the L trials.

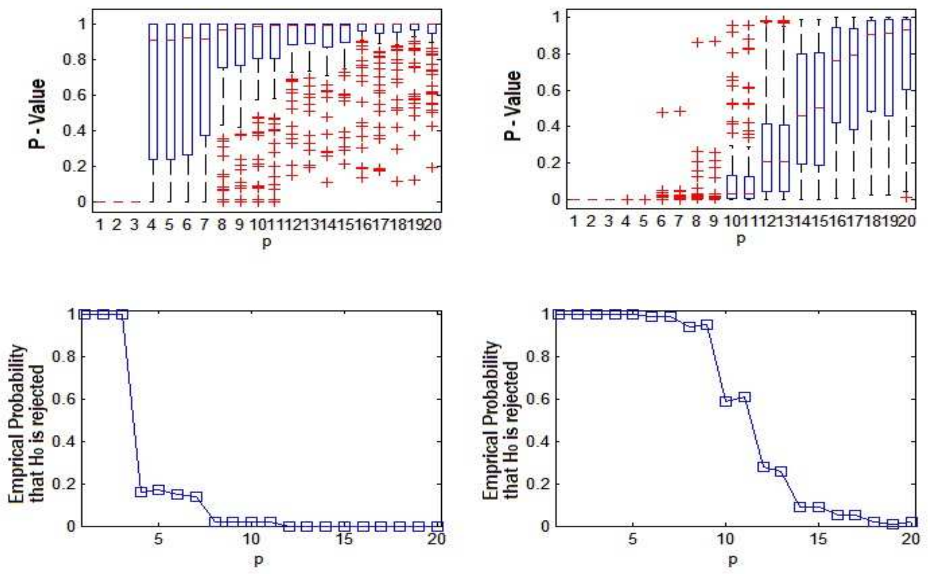


Figure 45: Plot of Neyman test result with $\hat{a}_k$ calculated by (60) in Example 2, Section 4.8.3. The two columns correspond to testing the two systems $m_1^*(\cdot)$ (true polynomial) and $m_2^*(\cdot)$ (inverse tangent). The two rows show the boxplot of the P-Value corresponds to testing hypothesis of different orders, as well as the empirical probability of rejecting null hypothesis based on the L trials.

of any orders for the nonlinear characteristics can be correctly tested with high probability. The power of this test, will be future work in this research. Again, similar techniques can also be applied to test other parametric assumptions for the nonlinear subsystem. Besides, orthogonal series analysis with coefficient shrinkage method, or with truncation parameter selection method in system identification are also examined.

It is worth mentioning that the methods used in Section 4 can also be extended to MISO systems. Hence, the coefficients shrinkage method can eliminate redundant coefficients for MISO system estimation, yielding, therefore, higher accuracy and overcome the curse of dimensionality.

5 Lasso Regression for Transient Stability Analysis

The previous sections have been focused on some theoretical issues about system modeling. In the following section, we are going to examine a system modeling problem in the context of transient stability applications.

5.1 Introduction to Transient Stability Analysis Using Lasso Regression

The concept of power system security is an indication of the power systems in the presence of disturbance. It is a time-varying property, and relevant research can be divided into several different fields. See Fig. 46 [34].

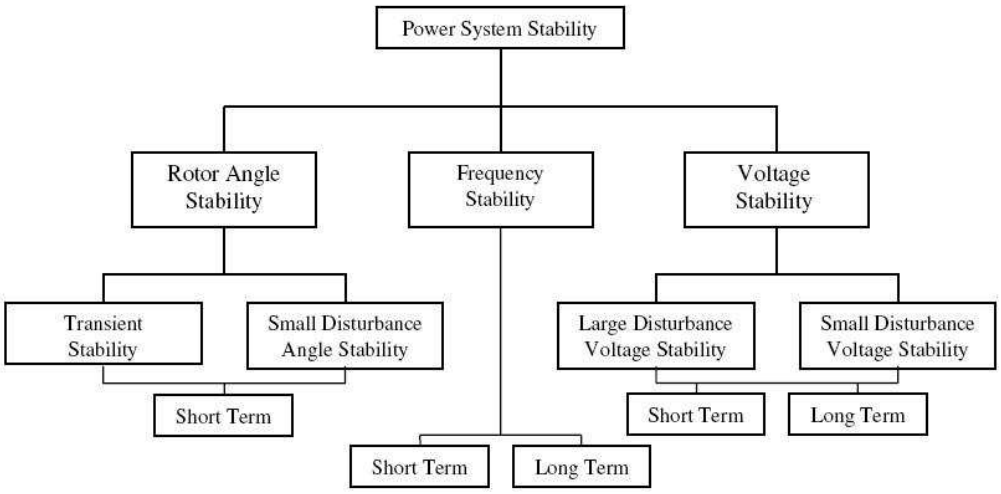


Figure 46: Classification of power system stability.

Transient stability is the ability of a power system to return to its normal operating condition when disturbances occur due to a fault in the system such as loss of a large load or loss of a generator. It is a reflection of the capability of the power system to absorb kinetic energy due to the transient disturbance. The transient stability behavior of a power system, in general, is determined by the steady state before disturbance, the nature of the fault, and the post-contingency structure of the power system. Hence, for a certain contingency in a given power system, transient stability is characterized only by the pre-contingency conditions.

The transient stability index (TSI), therefore, is only a function of initial operating point, given that a certain fault is being examined. i.e., TSI $= f_{TSI}(\mathbf{x})$, where $\mathbf{x} = (x_1, \cdots, x_p)$ is a random variable describing the pre-contingency state. In our studies, the fault critical clearing time (CCT) is used to represent the TSI. The transient stable region is characterized by TSI $\geq$ TSI$_0$, where TSI$_0$ is the threshold value. The boundary region $\{\mathbf{x} : f_{TSI}(\mathbf{x}) =$ TSI$_0\}$ is called the transient stability boundary (TSB), and it defines the boundary between secure and insecure regions of the initial state for power operating.

For a given initial state, the post-contingency transient stability behavior after a certain fault can be determined by solving a large number of coupled nonlinear differential and algebraic equations (DAE). Time-domain simulations for transient stability can be performed in such way to obtain the TSB. Meanwhile an alternative approach is to implement regression or pattern classification techniques. Comparing with time-domain simulation, regression/pattern classification methods has naturally its advantage in respect of speed, and is easily applicable in real scenarios when immediate decisions are necessary.

Among the existing literatures on regression analysis approach to the problem of transient stability, the Ridge Regression strategy has been proposed. This method offers some advantage over ordinary least squares in terms of lower prediction error. Nevertheless. the Ridge Regression algorithm is not able to eliminate redundant parameters, and as a result suffers the curse of dimensionality.

In this section we examine a modern regression technique for the transient stability problem that utilizes the l_1 penalty and is able to simultaneously estimate parameters of the models and eliminate redundant residuals. This algorithm is often referred to as the Lasso (Least Absolute Shrinkage and Selection Operator) method and is based on the concept of minimizing the predictive least square error with constraint to the weights penalized in l_1 norm [35] rather than l_2 norm in the Ridge Regression case. The Lasso method can eliminate or downweight input variables which are of only minor influence on the prediction error. The essential parameters of the model are estimated with the optimal parameteric accuracy. Thus by eliminating unnecessary input variables the Lasso algorithm can also achieve feature selection at the same time, thus provide important information about the mechanisms of the power system.

In the field of transient stability, previous works have been utilizing Ridge Regression, Kernel Ridge Regression methods, Support Vector Machines [34], [36], [37]. It is shown in this thesis that using Lasso regression achieves much favorable prediction in terms of the prediction error and moreover the capacity to simultaneously eliminate the majority of unnecessary features [35], [38].

5.2 Lasso Regression

5.2.1 From Ridge Regression to Lasso Regression

Transient stability data is usually high dimensional since measurement is taken from a large number of ports in the power system. Generally we are given data set of the form

$$\Big(\mathbf{X}_1, Y_1\Big), ..., \Big(\mathbf{X}_n, Y_n\Big),$$

with p-dimensional observations $\mathbf{X}_i = \Big(X_i^{(1)}, \cdots, X_i^{(p)}\Big)^T$ and 1-dimensional response Y_i. We wish to determine the new response when a new observation is given.

If we assume linear relationship between the response variable Y and the explanatory variable $X^{(j)}$s, then we have the linear regression model:

$$Y_i = \sum_{j=1}^{p} \beta_j X_i^{(j)} + \varepsilon_i \; (i = 1, \cdots, n), \tag{64}$$

where ε_i is the i.i.d noise with zero mean and independent of $\{\mathbf{X}_i\}$, and $\beta = \Big(\beta_1, \cdots, \beta_p\Big)$ is the vector of unknown parameters. The linear regression model is important because many other forms of regression (This includes, e.g., generalized linear methods) can be derived from the formula of linear regression.

Linear regression is also sometimes expressed in the matrix form:

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon, \tag{65}$$

with design matrix $\mathbf{X}_{n\times p} = \Big[\mathbf{X}_1, \cdots, \mathbf{X}_n\Big]^T$, response vector $\mathbf{Y}_{n\times 1} = \Big[Y_1, \cdots, Y_n\Big]^T$, parameter vector $\beta_{p\times 1} = \Big[\beta_1, \cdots, \beta_p\Big]^T$ and error vector $\varepsilon_{n\times 1} = \Big[\varepsilon_1, \cdots, \varepsilon_n\Big]^T$.

The explanatory variables $\mathbf{X}$ and the response variable Y are used after initial preprocessing so that they are with zero mean and standard deviation equal to one.

The classical way to estimate the model parameters β is to apply the least squares strategy, i.e., to minimize

$$\frac{1}{n}||\mathbf{Y} - \mathbf{X}\beta||_2^2 \tag{66}$$

This yields the well known least square solution $\hat{\beta}_{LS} = \Big(\mathbf{X}^T\mathbf{X}\Big)^{-1}\Big(\mathbf{X}^T\mathbf{Y}\Big)$.

The problem with the least square solution is its lack of stability, i.e., a small perturbation on data may yield a large variation on the model.

To address this issue of lack of stability, a Ridge Regression is based on the minimization of least squares with l_2 constraint to the weights:

$$||\mathbf{Y} - \mathbf{X}\beta||_2^2/n + \lambda||\beta||_2^2.$$

The solution to the Ridge Regression $\hat{\beta}_{Ridge}$ is the minimizer of the above criterion.

$$\hat{\beta}_{Ridge}(\lambda) = \arg\min_{\beta}\Big(||\mathbf{Y} - \mathbf{X}\beta||_2^2/n + \lambda||\beta||_2^2\Big) \tag{67}$$

The required minimization problem can also be written in the following equivalent primal form:

$$\hat{\beta}_{Ridge;primal}(s) = \arg\min_{\beta;||\beta||_2\leq s}\Big(||\mathbf{Y} - \mathbf{X}\beta||_2^2/n\Big).$$

Different from the Ridge Regression, Lasso regression uses the constraint in the form of the l_1 penalty:

$$\hat{\beta}_{Lasso}(\lambda) = \arg\min_{\beta} \Big(||\mathbf{Y} - \mathbf{X}\beta||_2^2/n + \lambda||\beta||_1 \Big). \tag{68}$$

The equivalent primal form of the minimization is the following:

$$\hat{\beta}_{Lasso;primal}(s) = \arg \min_{\beta;||\beta||_1 \le s} \Big(||\mathbf{Y} - \mathbf{X}\beta||_2^2/n \Big).$$

The comparison between Ridge Regression and Lasso regression can be illustrated by the contour lines of the sum of squares $||\mathbf{Y} - \mathbf{X}^T\mathbf{X}||_2^2$, shown in Fig. 5.2.1.

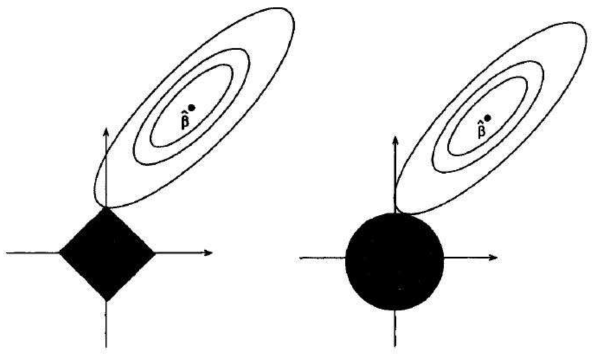


Figure 47: Contour lines of the sum of squares of regression. Left: Lasso method. Right: Ridge method.

In Fig. 5.2.1, the shaded area corresponds to the constraints imposed on the weights. The l_1 constraint corresponds to the square shape, therefore minimum sum of squares could happen at the corner which means one observation feature is to be set with zero weight. On the other hand the minimum sum of squares never happen at such corner places in case of the l_2 constraint which corresponds to Ridge Regression. This illustrate how Lasso method can achieve feature selection during the estimation process.

In order to achieve the required optimal variable selection and parameter estimation, one needs to select the regularization parameter λ. The penalty parameter λ determines the number of nonzero weights in the final solution of the regression problem. Standard approaches to specify λ are based on some versions of Cross-Validation re-sampling techniques. With the selected regularization parameter λ, the weights are calculated through certain algorithms, i.e. Shooting algorithm for Lasso (See section 5.2.3).

5.2.2 The Adaptive Lasso

For regression problems with high-dimensional data observations, the previously introduced Lasso method may not be enough to eliminate all the un-useful features. In order to eliminate more explanatory variables with less influence to the model, one might need to use the so-called "adaptive Lasso" [39], which is a re-weighted version of Lasso:

$$\hat{\beta}_{adapt}(\lambda) = \arg\min_{\beta} \left(||\mathbf{Y} - \mathbf{X}\beta||_2^2/n + \lambda \sum_{j=1}^{p} \frac{\beta_j}{\hat{\beta}_{init,j}} \right) \tag{69}$$

where $\hat{\beta}_{init}$ is an initial estimator for the weights, and it also has to qualify:

$$\hat{\beta}_{init,j} = 0 \;\Rightarrow\; \hat{\beta}_{adapt,j} = 0$$

The essence of the above adaptive Lasso approach is that greater penalty is assigned to the features with smaller weights in the first step.

Similarly, multi-step Lasso adaptive (MSA-Lasso) regression can be performed. This would eliminate further more features compared with Lasso regression or the two-step adaptive Lasso regression.

5.2.3 Shooting Algorithm for Lasso

In calculating the Lasso regression, one algorithm is the coordinate descent minimization. First, denote the criterion function by:

$$Q_\lambda(\beta) = \left(||\mathbf{Y} - \mathbf{X}\beta||_2^2/n + \lambda||\beta||_1 \right).$$

The gradient of $||\mathbf{Y} - \mathbf{X}\beta||_2^2/n$ is :

$$G_j(\beta) = -2\mathbf{X}_j^T(\mathbf{Y} - \mathbf{X}\beta)/n.$$

1. Set m=0. Let $\beta^{(0)} \in \mathbb{R}^p$ be an initial parameter estimate.

2. repeat

    - $m = m + 1$;
    - For $j = 1, \cdots, p$:

$$\beta_j^{(m)} = \frac{sign(Z_j)(|Z_j| - \frac{\lambda}{2})_+}{\hat{\Sigma}_{jj}},$$

$$Z_j = \mathbf{X}_j^T(\mathbf{Y} - \mathbf{X}\beta_{-j})/n,\ \hat{\Sigma} = n^{-1}\mathbf{X}^T\mathbf{X}.$$

3. until numerical convergence.

5.3 Generation of Transient Stability Data

The power system where regression estimate is performed is a medium scale real power system with 470 buses, as is shown in Fig. 5.3. It consists 470 buses, 45 generating units, 214 loads and 482 transmission lines, 152 fixed shunts, and 374 adjustable transformers [34]. All 45 generators are modeled with a 5th order generator modeled while the excitation systems of most generators are model with terminal voltage transducers, voltage regulators, exciters, and power system stabilizers. The original 470 Bus System was lightly loaded so that the system was very stable. In order to study the 470 Bus System under heavy load and generation conditions, a new base case was generated by increasing load and generation levels. The contingency is due to a 3-phase fault near bus 1007 on line 1007-1028 for 8 cycles and then the fault is cleared by opening line 1007-1028. This contingency is an example showing a case that the instability of the system is due to the swing of one generator against the rest of the system. Therefore for this contingency, the stability boundary is at 8 cycles. This study examines the cases that a perturbation of $\pm 15\%$ and $\pm 25\%$ for active and reactive power happens, and in these cases the perturbation for generator reference voltage setting is $\pm 2\%$.

Measurements are taken at the 470 buses. Therefore the observations are 939-dimensional, where 470 of them correspond to voltages and 469 of them are measurement of angles. The

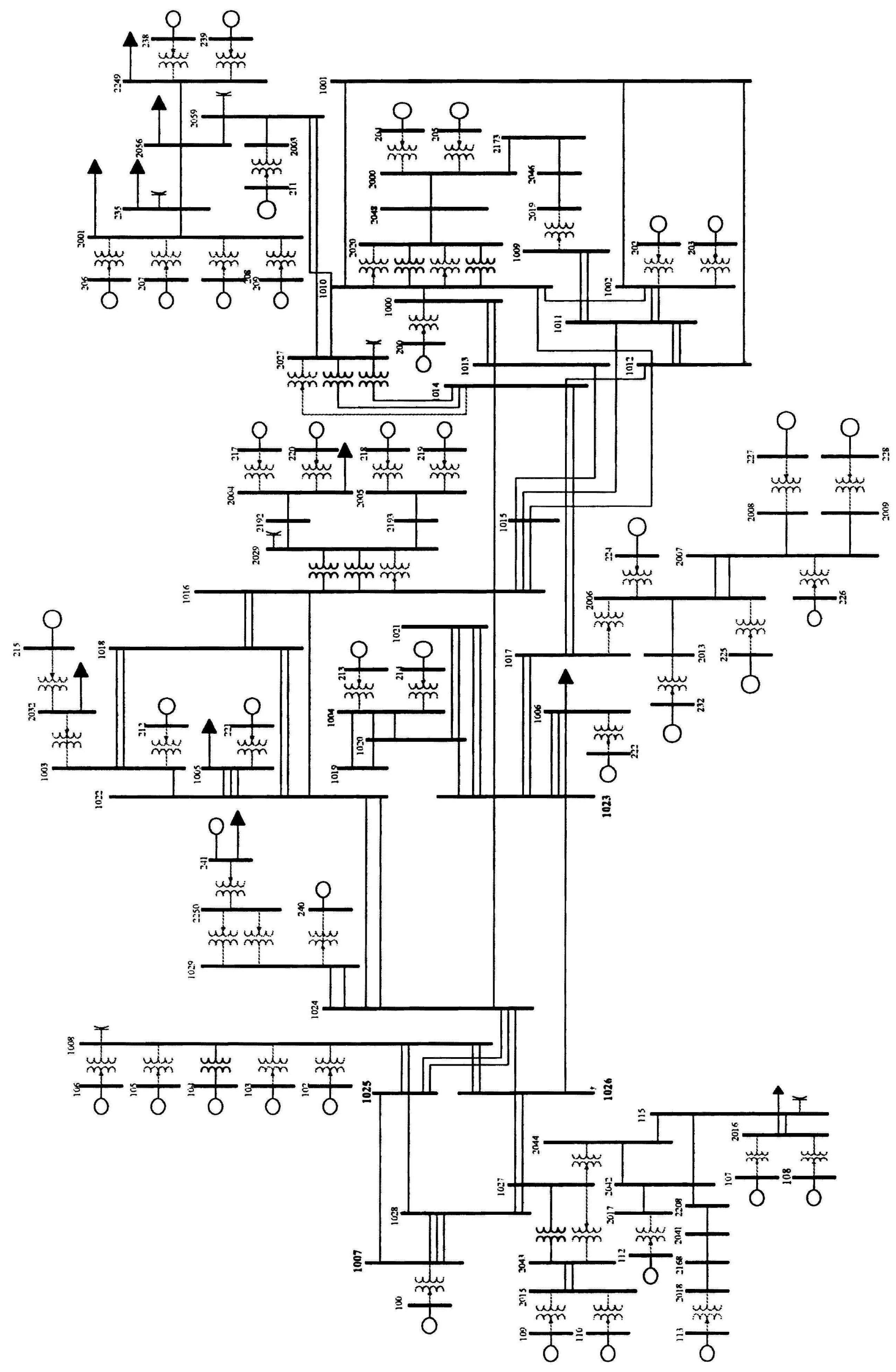


Figure 48: Single Line Diagram of Generators and 345 kV Network of 470 bus system.

per unit voltage vales and angles in radians are used to represent the input variable spaces. For each of the observation $\mathbf{X_i}$, Critical Clearing Time (CCT) is simulated as the response Y_i to the $i - th$ observation data $\mathbf{X_i}$. Here the CCT is used as a transient stability index.

In each cases, a training set of size $l = 800$ is examined. Independent data sets are used to evaluate the prediction errors. Data $\{(\mathbf{X}_1, Y_1), ..., (\mathbf{X}_n, Y_n)\}$ are first normalized so that the observations as the responses would have zero mean and unit standard deviation. Denote the normalized data set by $\{(\mathbf{X}_1^*, Y_1^*), ..., (\mathbf{X}_n^*, Y_n^*)\}$. Prediction errors are obtained in the form of mean square error for normalized responses:

$$\text{MSE(normalized)} = \sqrt{\frac{1}{l'}\sum_{i=1}^{l'}(Y_i^* - \hat{Y}_i^*)^2},$$

where Y_i^* is the normalized observation and $\hat{Y}_i^*$ is the predicted value, and l' is the length of the testing set which is independent from the training data set.

We also evaluate the root-mean-square-error(RMSE) for the original response:

$$\text{RMSE} = \sqrt{\frac{1}{l'}\sum_{i=1}^{l'}(Y_i - \hat{Y}_i)^2},$$

which exhibits actual physical meanings for application purpose. The unit of this error is "cycles".

Since we know the fault clearing time is 8 cycles in these simulations, the observed CCT can determine whether the initial contingency would lead to stable or unstable operating point. Therefore given a certain contingency, it is our interest to predict whether a certain pre-contingency state characterized by the 939-dimensional observation will lead to "stable" or "unstable" operating point. Thus we also use the following indices to evaluate the performance of the regression analysis:

- $FA = \frac{\sum(\text{False Alarms})}{l'}$

  A False Alarm occurs when an stable operating point (An pre-contingency operating point that leads to a transiently stable power system when subjected to a given contingency) is classified as unstable.

- $FD = \frac{\sum(\text{False Dismissals})}{l'}$

  A False Dismissal occurs when a stable operating point (An pre-contingency operating point that leads to a transiently unstable power system when subjected to a given contingency) is classified as stable.

- $FC = \frac{\sum(\text{False Alarms}+\text{False Dismissals})}{l'}$

  False Classification occurs when an unstable operating pint is classified as stable or stable operating point is classified as unstable.

- FD Range

  A False Dismissal means an unstable case is dismissed as stable. The CCT for the cases that lie on the boundary is equal to the actual fault clearing time. The CCT of the most unstable case dismissed as stable gives an indication of how close the FD cases are to the boundary. The FD range expresses the distance from the worst FD to the transient stability boundary.

- FA Range

  A False Alarm means that a stable case is dismissed as unstable. The CCT of the most stable case dismissed as unstable gives an indication of how close the FA cases are to the boundary. The FA range expresses the distance from the worst FA to the transient stability boundary.

5.4 Regression Analysis for 25% Perturbation Data

As described in the last section, the data set is:

- Dimension $P = 939$

- Training set size: $l = 800$

Prior to applying Lasso regression to the linear model, data need to be normalized in order to make sure each feature corresponds to observation values with mean zero and unit standard deviation. The regularization parameter λ plays a decisive role in the accuracy and final number of selected features. From (68), by intuition bigger value of λ would set more features to zero weights. The selection of regularization parameter λ requires certain kind of re-sampling techniques. Due to the consideration for computational complexity, the m-fold Cross-Validation could be suitable for such purpose.

Shooting algorithm in Section 5.2.3 for calculating lasso suggests that, with a certain selected regularization parameter λ, the computation complexity for calculating weights for a set of data of length n is:

$$(L+2)\cdot O\Big(nP^2\Big), \tag{70}$$

where P is the number of total features in the observation, or the dimension of one observation. L is the number of loops before the iteration stops.

The computation that takes most of the running time is the Cross-Validation that choose the optimal regularization parameter λ. In order to find optimal λ, suppose we need to compare the performance of N_λ candidate values of λ on average. For the m-fold Cross-Validation, we do calculations on a data set of approximately size $\frac{m-1}{m}\cdot n$ for m times. Suppose the "shooting algorithm" loop is to run T_1 times before it stops. Then the computational complexity is:

$$m\cdot(T_1+2)\cdot N_\lambda\cdot O\Big((\frac{m-1}{m}\cdot n)\cdot P^2\Big).$$

That is:

$$(m-1)(T_1+2)N_\lambda\cdot O\Big(nP^2\Big). \tag{71}$$

Once the regularization parameter λ has been selected, the regression complexity for calculating the coefficients is only:

$$(T_2+2)\cdot O\Big(nP^2\Big), \tag{72}$$

where T_2 is the number of running loops for “shooting algorithm” calculation. This is usually different from T_1, since calculating regression requires iterative optimization in “shooting algorithm” to be precisely converging, while this is not necessary for selecting optimal regularization parameter. On the other hand, using a large value T_1 may not be possible due to computational considerations.

In the following Section 5.4.1 and Section 5.4.2, Lasso regression are going to be conducted for linear models. Section 5.4.3 will conduct regression on the same data by Ridge Regression method, and the result will be used as a comparison for Lasso regression. Section 5.4.4 will show Lasso regression on extended models which include some reflection of quadratic relationships, and Section 5.4.5 will perform Kernel Ridge Regression, which is also a Ridge version of regression for higher-order model, and the result will also be compared with Lasso. Note that 5.4.1, Section 5.4.2, Section 5.4.4 have been using very parsimonious parameters (m, T_1, T_2). This is from the prospective of saving computational time. In Section 5.4.6 and Section 5.4.7, the role of these parameters is going to be discussed. Section 5.4.8 (and its quadratic extended models in Section 5.4.8) will give an example about Lasso regression under other parameter set-ups when more computational time is allowed. In the end, if Lasso regression under different parameter set-up can be compared, the best possible linear regression result will be shown in Section 5.4.9, and Section 5.4.9 is about the Lasso regression on its extended quadratic model.

In Section 5.4.7, it will be demonstrated by changing different fold of Cross-Validation in selecting regularization parameter in Ridge Regression, the prediction errors vary very little. Therefore the MSE for Ridge Regression shown in Section 5.4.3 and the MSE for Kernel Ridge Regression shown in Section 5.4.5 will be used as reference for comparison in Lasso regression of different set-ups.

5.4.1 Lasso Regression Analysis, Using 3-fold CV for Parameter Selection

According to (71), the fold of Cross-Validation is important for running time considerations. Here we use 3-fold Cross-Validation, and denote the selected λ by λ_{CV}. For the iteration times in the “shooting algorithm”, we use $T_1 = 2000$ and $T_2 = 15000$.

The mean square residual (MSR) for the constrained fit with constraint λ, and the nonzero weights number vs λ are shown in the Fig. 49. Note that λ_{CV} is emphasized as λ^* in the figure.

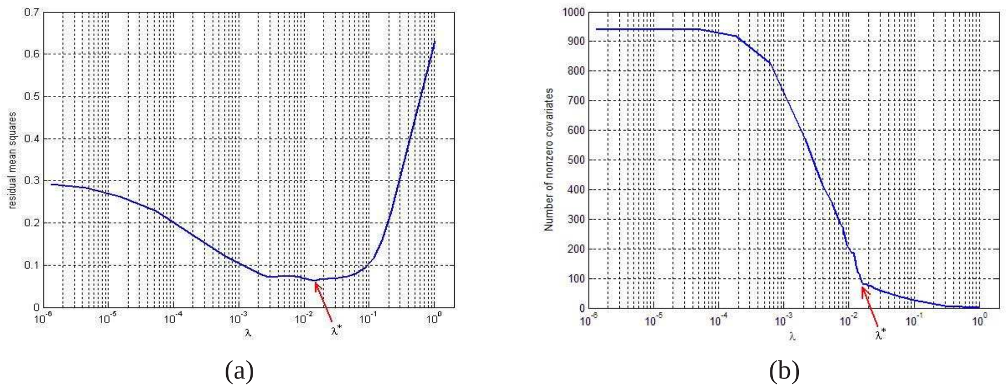


Figure 49: (a) Mean square residual (MSR) vs λ, (b) Number of selected features vs λ.

MSR is minimum (0.0627) when $\lambda = 0.0152$ and the result implies that 99 of the total 939 features are with nonzero weights. Then we use $\lambda^{(1)} = \lambda^*_{CV} = 0.0152$ to calculate the lasso regression weights for the whole training data set. Denote the estimated weights in this step by $\hat{\beta}^{(1)}$. The superscript here denotes it is the estimate in first-step Lasso. Then we find that the number of nonzero weights is 81. After applying linear model with weights $\hat{\beta}^{(1)}$ to the testing set and get $\{\hat{Y}_j^{(1)},\ j = 1, \cdots, 399\}$, we calculate the prediction error by MSE $= \frac{1}{l'} \sum_{j=1}^{l'} (Y_j - \hat{Y}_j^{(1)})^2$, where $l' = 399$ is the length of testing set. The result is 0.0735.

5.4.2 Multi-Step Lasso Regression

If we apply multi-step Lasso regression after getting the result in Section 5.4.1, we can get similar results after each step. It is shown in Table 7. Note that “# F” denote the number of non-zeros weights, which is also the selected features.

From here we come to the conclusion that by assuming linear model using the aforementioned procedure, the best achievable prediction error is 0.07082070 with the weights

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.01519734	0.06273982	0.07349168	0.86541573	81
2	0.00016132	0.05442373	0.07127725	0.85227782	53
3	0.00014077	0.05309322	0.07101012	0.85067927	45
4	0.00007339	0.05266391	0.07111103	0.85128347	43
5	0.00011722	0.05262231	0.07091939	0.85013561	43
6	0.00012935	0.05260251	0.07082070	0.84954386	43

Table 7: Multi-step Lasso regression for 25% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

obtained by Lasso regression method. Only 43 features are useful for the model.

As described in Section 5.3, the regression technique can be used for classification applications. The performance of Multi-step Lasso regression for classification of 25% disturbance data corresponding to the previous set-up is shown in Table 8.

It takes about 1.2 hours for a 2.30GHz computer to run the aforementioned regression. Computation time of Lasso regression with other selection of (m, T_1) can be calculated based on this result and (71).

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0.25%	4.76%	5.01%	7.188 - 8.000	8.000 - 8.929
2	0.25%	5.01%	5.23%	7.188 - 8.000	8.000 - 8.929
3	0.25%	5.51%	5.76%	7.188 - 8.000	8.000 - 8.929
4	0	5.01%	5.01%	8.000 - 8.000	8.000 - 8.929
5	0	5.01%	5.01%	8.000 - 8.000	8.000 - 8.929
6	0	5.26%	5.26%	8.000 - 8.000	8.000 - 8.929

Table 8: Classification indices for applying Multi-step Lasso regression for 25% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

5.4.3 Lasso Algorithm and Ridge Regression: Comparison Studies

Using the same training and testing sets of data, the previous Lasso regression performance is going to be compared with the Ridge Regression technique described by (67). Suppose we also use 3-fold Cross-Validation to select the regularization parameter in (67).

So the optimal λ for Ridge Regression is 3.7501. Denote this value by $\lambda_{Ridge,CV}$. By

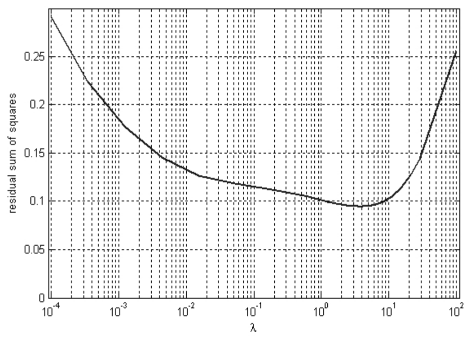


Figure 50: Mean square residual (MSR) vs λ in Cross-Validation by Ridge Regression.

using this regularization parameter and the whole training set, we calculate the weights using Ridge Regression. Then we use the testing set to evaluate the error. The mean square error of prediction corresponds to Ridge Regression is 0.0902079. The performance for regression as well as pattern classification is shown in Table 9 and Table 10.

Ridge	$\lambda_{Ridge,CV}$	MSR	MSE	RMSE(cycles)	# F
	3.75008570	0.09456501	0.0902079	0.95879995	939

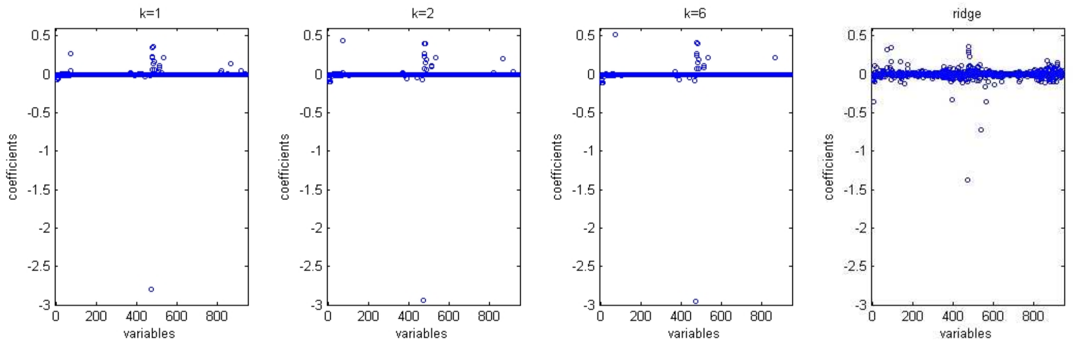
Table 9: Ridge Regression for 25% disturbance data. 3-fold CV used in selecting regularization parameter.

Ridge	FD	FA	FC	FD range (cycles)	FA range (cycles)
	0	4.26%	4.26%	8.000 - 8.000	8.000 - 9.772

Table 10: Classification indices for applying Ridge Regression. 3-fold CV used in selecting regularization parameter.

Comparing this result with the Lasso regression method, we notice that Lasso regression for all steps would achieve considerably smaller prediction mean square error, i.e. 6-th step Lasso leads to MSE 22.49% smaller than that of Ridge, or equivalently, RMSE 12.40% smaller. Also Lasso regression can automatically select as few as 43 features out of the 939, while Ridge Regression will lead to all weights to be nonzero, whether useful or not. The weights calculated by Lasso are shown in Fig. 51, aside with those calculated by Ridge method.

Ridge Regression shows smaller false alarm rates. But the False alarm range is much



(a) (b) (c) (d)

Figure 51: The weights determined by regression for 25% disturbance data. (a) One-step Lasso, (b) Two-step Lasso, (c) Six-step Lasso, (d) Ridge Regression.

bigger than Lasso regression with the aforementioned parameter setup.

It makes sense to compare the previous Lasso regression and Ridge Regression results since both of them are applied to linear models, both of them are based on minimization of mean square error with constraints on weights. In the following sections, models are extended to incorporate quadratic relations between the features. And Lasso regression on this extended model is going to be compared with Kernel Ridge Regression, which is an extension of Ridge Regression for higher-dimensional relationships.

5.4.4 Lasso Regression for Quadratic models

If we extend the linear model into quadratic model, that is, we assume the quadratic relationship between the observations and response, then the model is:

$$\mathbf{Y}^* = \tilde{\mathbf{X}}^* \tilde{\beta} + \varepsilon, \tag{73}$$

where $\mathbf{Y}^*{}_{n\times 1} = \left[Y_1^*, \cdots, Y_n^*\right]^T$, $\tilde{\mathbf{X}}^*_{n\times(p+3)p/2} = \left[\tilde{\mathbf{X}}_1^*, \cdots, \tilde{\mathbf{X}}_n^*\right]^T$,
$\tilde{\mathbf{X}}_i^* = [\mathbf{X}_i^{*(1)}, \mathbf{X}_i^{*(2)}, \cdots, \mathbf{X}_i^{*(p)}, \mathbf{X}_i^{*(1)}\mathbf{X}_i^{*(2)}, \mathbf{X}_i^{*(1)}\mathbf{X}_i^{*(3)}, \cdots, \mathbf{X}_i^{*(p-1)}\mathbf{X}_i^{*(p)}]$,
$\tilde{\beta}_{(p+3)p/2\times 1} = \left[\tilde{\beta}_1, \cdots, \tilde{\beta}_{(p+3)p/2}\right]^T$, $\varepsilon_{n\times 1} = \left[\varepsilon_1, \cdots, \varepsilon_n\right]^T$.

We can extend the data into quadratic forms to apply Lasso regression. But this means expanding 939 features in the linear model into 442269 covariates. According to (71), the complexity for computation would be $(442269/939)^2 = 221841$ times of the linear model, even without considering the necessity of using larger T_1 and T_2. Therefore it is too difficult for numerical analysis.

In order to solve this issue, we can try to approximate the model by expanding from the selected features. In the following part, Lasso regression will be applied to the models extended from the selected features of 2nd-step and 4th-step results shown in Table 7.

Lasso regression for quadratic model extended from 43 features From the selected features in 4-th step Lasso regression shown in Fig. 7, we can get $43 + 42 + \cdots + 1 = 946$ quadratic-term features. Together with the original 43 linear-term features, we form a model of 989 covariates. After applying normalization to each covariates, we conduct Lasso regression analysis on the new model. The settings in regression are selected as the following:

- Use 3-fold Cross-Validation to select optimal regularization parameter λ,
- $T_1 = 2000$,
- $T_2 = 15000$.

The result calculated by multi-step Lasso algorithm is shown in Table 11 and Table 12.

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00264484	0.04455195	0.05262265	0.73230509	88
2	9.3502e-05	0.03015355	0.04982496	0.71257272	54
3	2.8959e-05	0.02894741	0.05016505	0.71500049	51
4	9.8804e-06	0.02887865	0.05034889	0.71630943	51
5	1.0210e-05	0.02887702	0.05035593	0.71635947	51
6	1.0210e-05	0.02887689	0.05035731	0.71636934	51

Table 11: Multi-step Lasso regression for 25% disturbance data with 989 extended features. 3-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

So here we achieve better result than Lasso regression for the linear model. Finally, the 51 extended features are selected. By further checking these 51 extended features, we find

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0	1.75%	1.75%	8.000 - 8.000	8.000 - 8.823
2	1.00%	1.25%	2.26%	7.768 - 8.000	8.000 - 9.772
3	1.00%	1.25%	2.26%	7.768 - 8.000	8.000 - 9.772
4	1.00%	1.25%	2.26%	7.768 - 8.000	8.000 - 9.772
5	1.00%	1.25%	2.26%	7.768 - 8.000	8.000 - 9.772
6	1.00%	1.25%	2.26%	7.768 - 8.000	8.000 - 9.772

Table 12: Classification indices for applying Multi-step Lasso regression for 25% disturbance data with 989 extended features. 3-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

that among them, 50 are quadratic terms from 42 of the original linear features, and the other one is an linear feature. Overall, 42 features from the linear model have contributed to these selected 51 extended features. Here MSE is 0.05035731, which is equal to 55.80% of the Ridge Regression MSE in Section 5.4.3, or 62.14% of the Kernel Ridge Regression MSE, see Section 5.4.5.

Lasso regression for quadratic model extended from 53 features If we do feature expansion from the 53 features selected by the 2-step linear Lasso regression, and perform similar techniques, we can get the following results. Note that the total number of extended features in this case is 1484, and the settings for Lasso regression is similar to the last section:

- Use 3-fold Cross-Validation to select optimal regularization parameter λ,
- $T_1 = 2000$,
- $T_2 = 15000$.

Therefore we finally select 38 features from the 1484 quadratic features. All these 38 extended features are quadratic relationship between 33 of the original linear features. The MSE corresponds here is equal to 51.73% of the linear Ridge Regression MSE in Section 5.4.3, or 57.63% of the Kernel Ridge Regression MSE (See section 5.4.5).

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00147384	0.03962129	0.04816466	0.70059970	202
2	0.00041776	0.03497379	0.04710221	0.69282948	47
3	0.00015036	0.02892003	0.04749144	0.69568621	39
4	5.4202e-05	0.02877159	0.04696164	0.69179487	38
5	7.8577e-05	0.02881423	0.04671763	0.68999528	38
6	2.5137e-05	0.02882314	0.04669970	0.68986284	38

Table 13: Multi-step Lasso regression for extended model of 25% disturbance data with 1484 features. 3-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0.50%	1.00%	1.50%	7.716 - 8.000	8.000 - 8.718
2	0.50%	1.50%	2.01%	7.716 - 8.000	8.000 - 8.981
3	0.75%	1.00%	1.75%	7.716 - 8.000	8.000 - 9.772
4	0.75%	1.25%	2.01%	7.716 - 8.000	8.000 - 9.772
5	1.00%	1.25%	2.26%	7.188 - 8.000	8.000 - 9.772
6	1.00%	1.25%	2.26%	7.188 - 8.000	8.000 - 9.772

Table 14: Classification indices for applying Multi-step Lasso regression for 25% disturbance data with 1484 extended features. 3-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

5.4.5 Lasso and Kernel Ridge Regression: Comparison Studies

Kernel Ridge Regression is an extension from linear Ridge Regression. It is based on the following extension of the model:

$$y = \sum_{i=1}^{n} \alpha_i K(x_i, x) \tag{74}$$

where $K(\cdot, \cdot)$ is a kernel function which satisfies the so-called "Mercer's Theorem". Therefore kernel regression can reflect higher order relationship between features.

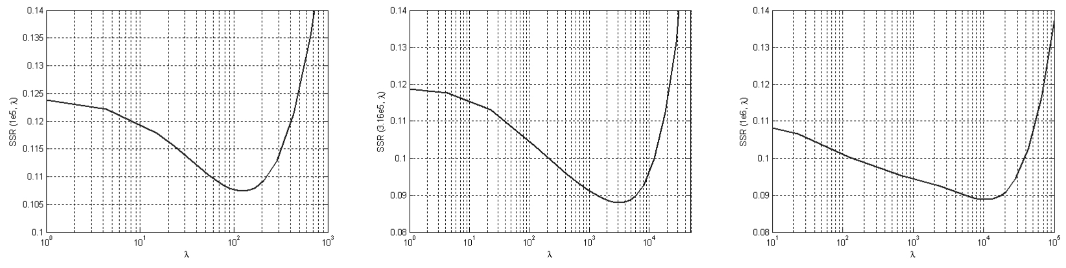
It can be shown that the solution to kernel regression can be expressed as:

$$\hat{\alpha}_i = (\mathbf{K} + n\lambda\mathbf{I})^{-1}\mathbf{y}, \tag{75}$$

where $\mathbf{K}$ is a $n \times n$ matrix with (i, j) element equal to $K(x_i, x_j)$, λ is the regularization parameter (similar to that in the linear Ridge Regression), and $\mathbf{I}$ is a $n \times n$ identity matrix.

Here we use the polynomial kernel $\mathbb{K}(\mathbf{a}, \mathbf{b}) = (q + \mathbf{a}^T\mathbf{b})^p$ here. We first examine the $p = 2$ case, which corresponds to the quadratic relationship between the features. (p=1 would be quadrivalent to linear Ridge Regression, which has benn already examined in the previous sections.) Again, we use the same training and testing data set, and 3-fold Cross-Validation is used to choose the optimal parameter q and λ.

We first choose different value for constant term q, and then plot the regression mean squares vs λ. The following figure is an illustration of the relationship between the mean square residual (MSR) and the regularization parameter λ when different value of q is adopted.



(a) (b) (c)

Figure 52: Regression sum of squares (MSR) vs regularization parameter λ under different value of q in Kernel Ridge Regression of order 2. (a) $q = 1 \times 10^5$, (b) $q = 3.16 \times 10^5$, (c) $q = 1 \times 10^6$.

The regression sum of squares (MSR) is a function of both q and λ in this case. We find that MSR is minimized when $(q, \lambda) = (316228, 3225)$. Then we fix $\lambda = 3225$ value and plot MSR vs q in Fig. 53 as an illustration of MSR changing with q.

Using $(q, \lambda) = (316228, 3225)$, we find the prediction error is 0.0851 for the Kernel Ridge Regression using polynomial kernel of order 2.

We notice that the Kernel Ridge Regression result is improved from the linear Ridge Regression. But still its MSE is significantly larger than the Lasso regression counterpart.

If we apply the same procedure using polynomial kernel of higher orders ($p \geq 3$), the result are shown in Table 33 and Table 16.

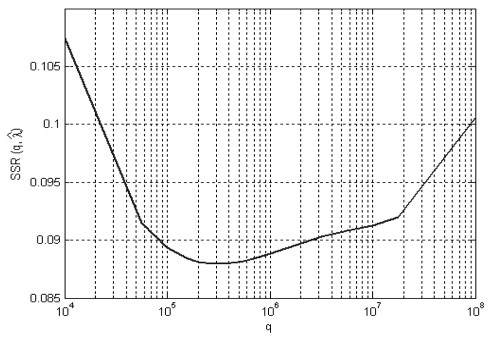


Figure 53: Regression sum of squares (MSR) vs parameter q when $\lambda = 3225$.

order p	$q_{3CV}^{(p)}$	$\lambda_{3CV}^{(p)}$	MSR	MSE	RMSE(cycles)	# F
2	316228	3224.9	0.08795147	0.08513388	0.93144437	939
3	9474635	1612079.2	0.10264472	0.08146376	0.91114595	939
4	14330126	5.06×10^{11}	0.10262463	0.08131985	0.91034078	939
5	18938420	0.01	0.10260855	0.08119592	0.92983016	939
6	23713737	0.01	0.10259889	0.08114725	0.91993648	939
⋮	⋮	⋮	⋮	⋮	⋮	⋮
12	52329912	0.01	0.10257793	0.08104084	0.90877776	939

Table 15: Kernel Ridge Regression of different orders for 25% disturbance data. 3-fold CV used in selecting regularization parameter.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
2	0.25%	3.76%	4.01%	7.188 - 8.000	8.000 - 9.192
3	0.75%	2.26%	3.01%	7.716 - 8.000	8.000 - 9.087
4	0.75%	2.26%	3.01%	7.716 - 8.000	8.000 - 9.772
5	0.25%	3.51%	2.76%	7.188 - 8.000	8.000 - 9.192
6	0.25%	4.01%	4.26%	7.188 - 8.000	8.000 - 9.772
⋮	⋮	⋮	⋮	⋮	⋮
12	0.75%	2.26%	3.01%	7.716 - 8.000	8.000 - 9.087

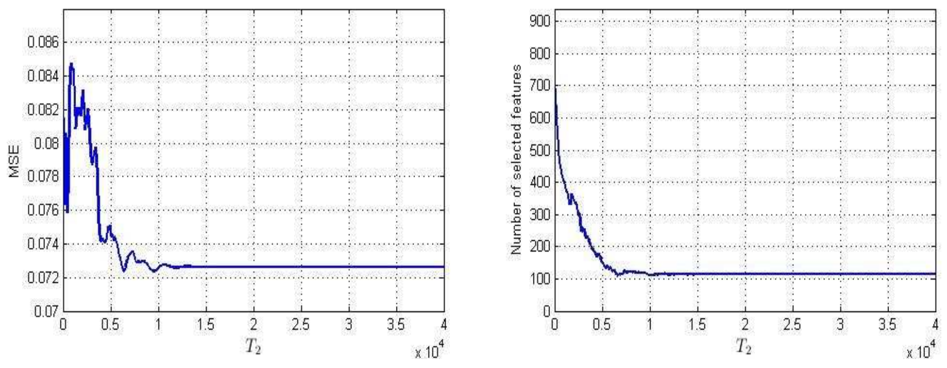
Table 16: Classification indices for applying Kernel Ridge Regression of different orders for 25% disturbance data. 3-fold CV used in selecting regularization parameter.

5.4.6 Selection of the Loop Parameter

As mentioned in the beginning of Section 5.2.3, the iteration times for minimization in the "shooting algorithm for Lasso" can have influence for the final regression result. Therefore they need to be practically taken into consideration. Once the regularization parameter λ is determined, T_2 will play a role of ensuring the convergence of the "shooting algorithm for Lasso". Numerical solution for T_2 might not be explicit. We conduct the following experiment for the 25% disturbance data of size 800 as mentioned in the previous sections.

- Regularization parameter λ is selected by 3-fold Cross-Validation. T_1=10000,
- With regard to first step Lasso regression, examine regression result while using different value of T_2.

The result is shown in Fig. 54. We come to the conclusion that for this type of data with dimension 939, "shooting algorithm" should run more than around 11000 loops to get a consistent prediction error, as well as an unchanged number of selected features.



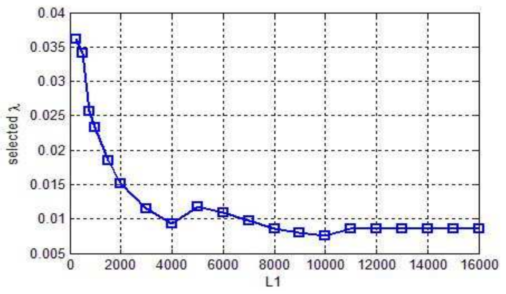
(a) (b)

Figure 54: (a). MSE vs T_2, (b) Number of selected features vs T_2.

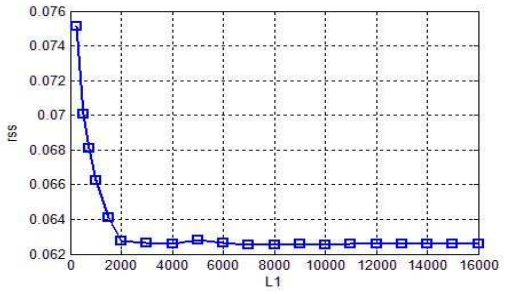
Although at least 11000 runs must be allowed for "shooting algorithm" to calculate precisely, optimization for selecting λ doesn't necessarily need as much as this number. This is due to the following reasons: Optimal λ could be found before T_2 times of iteration. Also, a smaller T_1 might sometimes lead to better performance.
The following experiment is conducted:

- Regularization parameter λ is selected by 3-fold Cross-Validation. $T_2 = 15000$. Based on the previous discussion, this corresponds to large enough iterations for the "shooting algorithm" loop to converge.
- Examine the regression result while using different values of T_1.

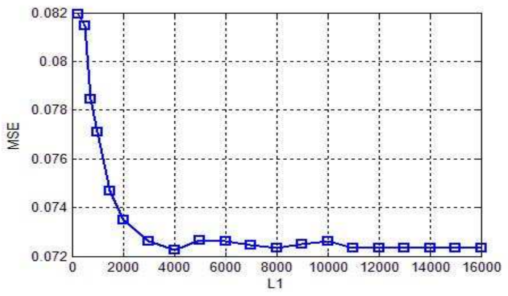
The result is shown in Fig. 55. We notice that after about 11000 times of iteration in



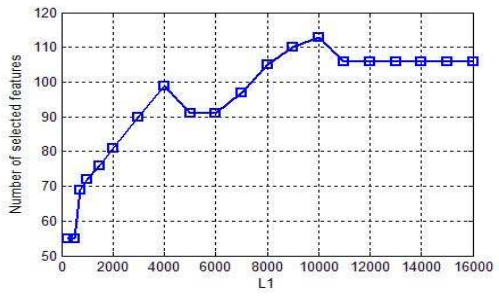
(a)



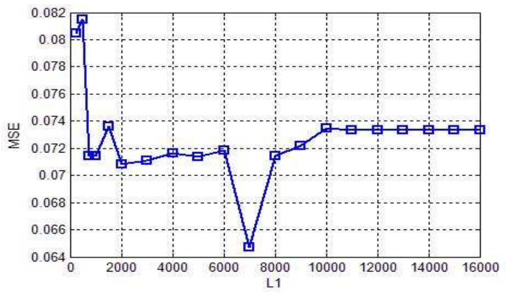
(b)



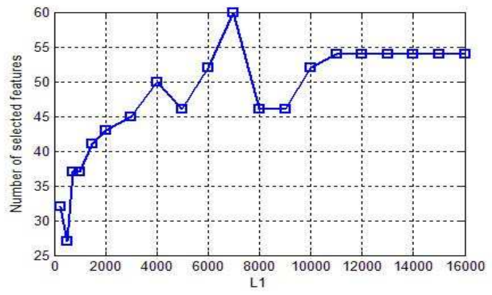
(c)



(d)



(e)



(f)

Figure 55: The influence of loop parameter T_1. (a) Value of selected $\lambda_{3CV}^{(1)}$ vs T_1, (b) Corresponding MSR vs T_1, (c) MSE for the 1-st step Lasso vs T_1, (d) Number of selected features in the 1-st Lasso vs T_1, (e) MSE for the 6-th step Lasso vs T_1, (f) Number of selected features in the 6-th step Lasso vs T_1.

the "shooting algorithm" loop, the result will tend to be the same, this re-confirms the conclusion from Fig. 54. However, using a smaller value for T_1 also provides a relatively good result in terms of regression error. It sometimes even lead to better results. In practice it would be a matter of luck for the selection of T_1, we only know as long as T_1 is greater than about 2000, there is a great chance of regression error close to the level that using

$T_1 > 11000$. Besides, using a smaller value for T_1 most of the time leads to smaller number of selected features than using $T_1 \geq 11000$, which is another desirable character for regression analysis.

About the 1-st step Lasso regression, among $T_1 \in \{3000, 4000, 5000, \cdots, 10000\}$. All values of T_1 will lead to mean square error within 99.86% to 103.73% of the MSE that $T_1 \geq 110000$ is used. And for the 6-th step Lasso regression result, we see that for $T_1 \in [750, 11000]$, the majority of T_1 values lead to surprisingly smaller MSE than when $T_1 \geq 110000$ is used, i.e. $T_1 = 7000$ has MSE that is 88.21% of the MSE that $T_1 \geq 110000$ is used.

5.4.7 Discussions about Re-sampling

We have performed Lasso regression on 25% disturbance data when different T_1 and different fold of Cross-Validation are used. It seems that the $5 - 10$ fold of Cross-Validation is usually better than 3 or 4 fold of Cross-Validation. Most of linear Lasso regression MSE using $5 \leq m \leq 10$ have regression MSE that is 75% - 80% of the MSE using $m = 3, T_1 = 2000$, the most parsimonious set-up (See Section 5.4.2). Under some other set-ups, the performance of Lasso regression could lead to further smaller MSE, See Section 5.4.9.

On the other hand, different fold of Cross-Validation doesn't change the performance of Ridge Regression very much. See Table 17 and Table 18. Therefore the Ridge Regression in Section 5.4.3, and the Kernel Ridge Regression in Section 5.4.5, can represent the performance of the two algorithms in general.

5.4.8 Lasso Regression Analysis, 7-fold CV for Parameter Selection

Multi-Step Lass regression on linear model The aforementioned Lasso regression uses 3-fold of Cross-Validation. Here we examine another parameter set-up:

- Use 7-fold Cross-Validation to select regularization parameter λ,

m -fold of CV	$\lambda^{(m)}$	MSR	MSE	RMSE(cycles)	# F
3	3.75008570	0.09456500	0.09020790	0.95879995	939
4	3.51406394	0.08903923	0.09022665	0.95889957	939
5	2.91280277	0.08997186	0.09038246	0.95972717	939
6	3.49257955	0.08901096	0.09022890	0.95891154	939
7	0.07459192	0.08823106	0.08775725	0.94568659	939
8	0.08380993	0.08441819	0.08798486	0.94691216	939
9	0.07269519	0.08288871	0.08771057	0.94543501	939
10	0.06630639	0.08272571	0.08757316	0.94469416	939

Table 17: Ridge Regression for 25% disturbance data. Different fold of Cross-Validation re-sampling technique examined.

m -fold of CV	FD	FA	FC	FD range (cycles)	FA range (cycles)
3	0.00%	4.26%	4.26%	8.000 - 8.000	8.000 - 9.772
4	0.00%	4.26%	4.26%	8.000 - 8.000	8.000 - 9.772
5	0.00%	4.51%	4.51%	8.000 - 8.000	8.000 - 9.772
6	0.00%	4.26%	4.26%	8.000 - 8.000	8.000 - 9.772
7	0.75%	1.75%	2.51%	7.716 - 8.000	8.000 - 9.192
8	0.75%	2.01%	2.76%	7.716 - 8.000	8.000 - 9.192
9	0.75%	1.75%	2.51%	7.716 - 8.000	8.000 - 9.192
10	0.75%	1.50%	2.26%	7.716 - 8.000	8.000 - 9.192

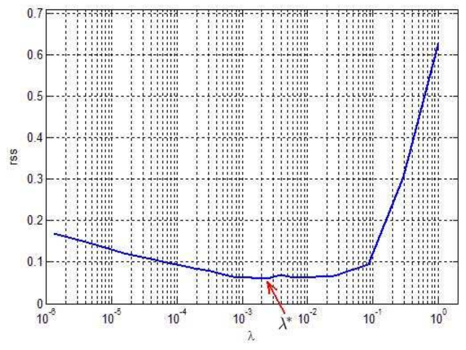
Table 18: Classification indices for applying Ridge Regression to 25% disturbance data. Different fold of Cross-Validation re-sampling technique examined.

- $T_1 = 10000$,
- $T_2 = 15000$.

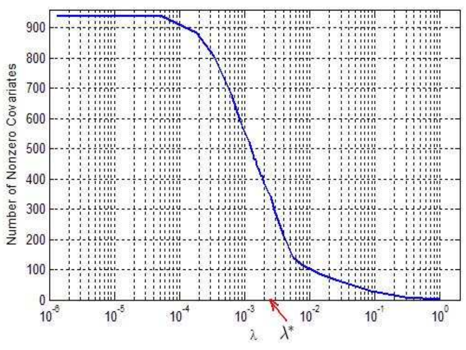
According to (71) and Formula 72, this set-up would take 15 times processing time as the Lasso linear regression in Section 5.4.1 and Section 5.4.2.

For the 1-st step Lasso regression, the mean square residual (MSR) for the constrained fit with constraint λ, and the nonzero weights number vs λ are shown in the Fig. 56. Note that the optimal λ is emphasized as in the figure.

MSR is minimum (0.0589) when $\lambda = 0.00162$. Then we use $\lambda^{(1)} = \lambda^*_{CV} = 0.00162$ to calculate the Lasso regression weights for the whole training data set. Denote the estimated weights in this step by $\hat{\beta}^{(1)}$. The superscript here denotes it is the estimate in first-step Lasso. Then we find that the number of nonzero weights is 378. We apply the linear model



(a)



(b)

Figure 56: (a) Mean squre residual (MSR) vs λ, (b) Number of selected features vs λ.

with weights $\hat{\beta}^{(1)}$ to the testing set and get $\{\hat{Y}_j^{(1)},\ j = 1, \cdots, 399\}$. We calculate the prediction error by MSE $= \frac{1}{l'} \sum_{j=1}^{l'} (Y_j - \hat{Y}_j^{(1)})^2$, where $l' = 399$ is the length of testing set. The result is 0.066285.

Then by applying adaptive Lasso and multi-step Lasso regression based on this result, we get results shown in Table 19. Again, "# F" denotes the number of non-zeros weights, which is also the selected features.

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00162370	0.05891837	0.06628522	0.82189068	378
2	0.00035988	0.03728863	0.05514453	0.74964724	79
3	7.77325e-9	0.0335196	0.05692402	0.76164657	79
4	0.00004713	0.03328281	0.05636768	0.75791551	74
5	0.00003339	0.03311114	0.05645032	0.75847090	73
6	1.55088e-8	0.03303029	0.05707699	0.76266927	73

Table 19: Multi-step Lasso regression for 25% disturbance data. 7-fold CV used in selecting regularization parameter, $T_1 = 10000$ and $T_2 = 15000$.

The best achievable prediction mean square error here is 0.05515. Only $73 - 79$ features are useful for the model compared with the total 939 features in the model. Note that this performance has mean square error 22.33% smaller than the result shown in Section 5.4.2.

The pattern classification performance is also examined. It is shown in Table 20.

Lasso regression for quadratic models, with features extended from the linear model

In this section we extend model to include quadratic relationship between the 79 features

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0	2.76%	2.76%	8.000 - 8.000	8.000 - 9.192
2	0	1.25%	1.25%	8.000 - 8.000	8.000 - 8.348
3	0	0	0	8.000 - 8.000	8.000 - 8.000
4	0	0.05%	0.05%	8.000 - 8.000	8.000 - 8.348
5	0	0.05%	0.05%	8.000 - 8.000	8.000 - 8.348
6	0	0.05%	0.05%	8.000 - 8.000	8.000 - 8.348

Table 20: Classification indices for applying Multi-step Lasso regression for 25% disturbance data. 7-fold CV used in selecting regularization parameter, $T_1 = 10000$ and $T_2 = 15000$.

selected by the 2-nd step Lasso regression result shown in Table 19. The extended model is going to have $79 + 78 + \cdots + 1$ quadratic terms, as well as the 79 linear terms. In total, that is 3239 extended features.

The set-up is:

- Use 3-fold Cross-Validation to select regularization parameter λ,
- $T_1 = 4000$,
- $T_2 = 40000$.

We use 3-fold Cross-Validation for considerations of computational complexity. According to (71), the computation would cost about 21.5 times of that the simulation in Section 5.4.1 takes. The result for multi-step Lasso regression is shown in Table 21.

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00124000	0.04406797	0.05289462	0.73419511	300
2	0.00015648	0.03161252	0.05438718	0.74448164	88
3	0.00003655	0.02595700	0.05009815	0.71452358	75
4	0.00001880	0.02530710	0.04938512	0.70942054	73
5	0.00002336	0.02515905	0.04920196	0.70810375	70
6	0.00000513	0.02508208	0.04928440	0.70869676	70

Table 21: Multi-step Lasso regression for extended model of 25% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 4000$ and $T_2 = 40000$.

The best achievable prediction mean square error here is 0.04928. Only 70 out of the total 3239 extended features are selected. Comparing with the Kernel Ridge Regression results, the 5-th step Lasso regression has mean square error 39.04% smaller, or equivalently, the RMSE 22.08% smaller.

Interestingly, if we examine the 300 extended features selected by the 1-step Lasso, they are from the cross-relationship of 78 original features. The 88 selected extended features in step 2 are from 62 of the original features, and the selected features from step 3 to step 6 are all from 59 of the original features.

The pattern classification performance is also examined. It is shown in Table 22.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0.5%	1.25%	1.75%	7.188 - 8.000	8.000 - 9.772
2	0.74%	1.00%	1.75%	7.716 - 8.000	8.000 - 9.772
3	1.00%	1.00%	2.01%	7.188 - 8.000	8.000 - 8.876
4	1.00%	1.00%	2.01%	7.188 - 8.000	8.000 - 8.876
5	1.00%	1.00%	2.01%	7.188 - 8.000	8.000 - 8.876
6	1.00%	1.25%	2.26%	7.188 - 8.000	8.000 - 9.772

Table 22: Classification indices for applying Multi-step Lasso regression to extended model of 25% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 4000$ and $T_2 = 40000$.

5.4.9 Lasso Regression Analysis, 6-fold CV for Parameter Selection

Multi-step Lass regression on linear model Section 5.4.1 and 5.4.2 represent the Lasso linear regression that uses almost the most time-saving set-up, while Section 5.4.8 can represent a random setup of m and T_1 for Lasso regression on the linear model. In Section 5.4.9 we are going to show the Lasso regression analysis of the same data assuming linear model, under the setup that produces the best performance within the numerous situations we examined: We have examined the Lasso regression with $m = 2, 3, 4, \cdots, 10$ folds of Cross-Validation, and $L_1 = 2000, 3000, 4000, \cdots, 10000$ as the value of loops in "shooting Lasso" to select regularization parameter λ. The set-up that leads the smallest error is the following:

- Use 6-fold Cross-Validation to select regularization parameter λ,
- $T_1 = 4000$,

- $T_2 = 15000$.

According to (71) and (72), this set-up would take 5 times processing time as the Lasso linear regression in Section 5.4.1 and Section 5.4.2.

For the 1-st step Lasso regression, the Mean squre residual (MSR) for the constrained fit with constraint λ, and the nonzero weights number vs λ are shown in the Fig. 56. Note that λ^*_{CV} is emphasized as λ^* in the figure.

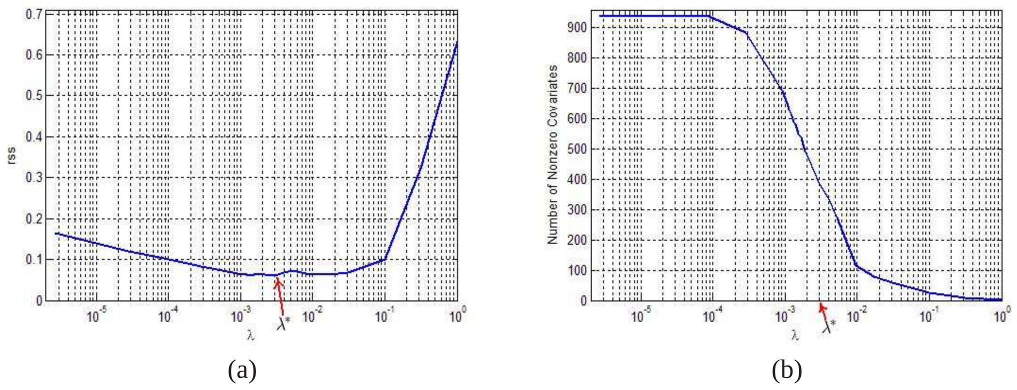


Figure 57: (a) Mean squre residual (MSR) vs λ, (b) Number of selected features vs λ.

MSR is minimum (0.0614) when $\lambda = 0.00129$. Then we use $\lambda^{(1)} = \lambda^* = 0.00129$ to calculate the lasso regression weights for the whole training data set. Denote the estimated weights in this step by $\hat{\beta}^{(1)}$. The superscript here denotes it is the estimate in first-step Lasso. Then we find that the number of nonzero weights is 434. We apply the linear model with weights $\hat{\beta}^{(1)}$ to the testing set and get $\{\hat{Y}^{(1)}_j,\ j = 1, \cdots, 399\}$. We calculate the prediction error by MSE $= \frac{1}{l'}\sum_{j=1}^{l'}(Y_j - \hat{Y}^{(1)}_j)^2$, where $l' = 399$, is the length of testing set. The result is 0.07037.

Then by applying adaptive Lasso and multi-step Lasso regression based on this result, we get results shown in Table 23. Again, "# F" denotes the number of non-zeros weights, which is also the selected features.

The best achievable prediction mean square error here is 0.05068. Only $32-42$ features are useful for the model compared with the total 939 features in the model. Note that this performance has mean square error 71.09% of the result shown in Section 5.4.2, or 55.81%

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00129189	0.06140993	0.07036502	0.84680641	434
2	0.00122544	0.03662763	0.05309092	0.73555613	59
3	0.00005680	0.03182768	0.05068121	0.71866947	45
4	0.00033748	0.03072977	0.05034812	0.71630393	33
5	0.00002738	0.03045304	0.05100691	0.72097506	32
6	0.00000017	0.03040617	0.05088550	0.72011644	32

Table 23: Multi-step Lasso regression for 25% disturbance data. 6-fold CV used in selecting regularization parameter, $T_1 = 4000$ and $T_2 = 15000$.

of the MSE using Ridge Regression.

The pattern classification performance is also examined. It is shown in Table 24.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0.00%	2.51%	2.51%	8.000 - 8.000	8.000 - 8.823
2	0.00%	1.50%	1.50%	8.000 - 8.000	8.000 - 8.401
3	0.00%	0.50%	0.50%	8.000 - 8.000	8.000 - 8.243
4	0.00%	0.50%	0.50%	8.000 - 8.000	8.000 - 8.243
5	0.00%	0.50%	0.50%	8.000 - 8.000	8.000 - 8.243
6	0.00%	0.50%	0.50%	8.000 - 8.000	8.000 - 8.243

Table 24: Classification indices for applying Multi-step Lasso regression for 25% disturbance data. 6-fold CV used in selecting regularization parameter, $T_1 = 4000$ and $T_2 = 15000$.

Lasso regression for quadratic models, with features extended from the linear model

Extended model from 33 features Similar to Section 5.4.4, here we extend model to include quadratic relationship between the 33 features selected by the 4-th step Lasso regression result shown in Table 23. The extended model is going to have $33 + 32 + \cdots + 1$ quadratic terms, as well as the 33 linear terms. In total, that is 594 extended features.

The set-up is:

- Use 3-fold Cross-Validation to select regularization parameter λ,
- $T_1 = 3000$,

- $T_2 = 20000$.

We use 3-fold Cross-Validation for considerations of computational complexity. The result for multi-step Lasso regression is shown in Table 25.

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00000313	0.04477467	0.03318769	0.58155957	594
2	0.00082748	0.02020760	0.03421298	0.59047450	438
3	0.00078259	0.01571584	0.03388333	0.58762299	352
4	0.00053177	0.01493181	0.03219415	0.57278838	312
5	0.00051229	0.01375866	0.03139655	0.56564855	278
6	0.00058708	0.01304152	0.03118869	0.56377303	253

Table 25: Multi-step Lasso regression for extended model of 25% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 3000$ and T_2=20000.

The best achievable prediction mean square error here is 0.03119. This error is only 34.68% of the MSE of Ridge Regression (See Section 5.4.3), or 38.49% of the MSE of Kernel Ridge Regression (See Section 5.4.5).

The pattern classification performance is also examined. It is shown in Table 26.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	1.00%	2.01%	3.01%	7.241 - 8.000	8.000 - 8.770
2	1.00%	1.75%	2.76%	7.241 - 8.000	8.000 - 8.718
3	0.50%	1.50%	2.01%	7.241 - 8.000	8.000 - 8.718
4	0.75%	1.75%	2.51%	7.241 - 8.000	8.000 - 8.718
5	1.00%	1.25%	2.26%	7.188 - 8.000	8.000 - 8.718
6	1.00%	1.75%	2.76%	7.188 - 8.000	8.000 - 8.718

Table 26: Classification indices for applying Multi-step Lasso regression to extended model of 25% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 3000$ and T_2=20000.

5.5 Regression Analysis for 15% Perturbation Data

Similar to the 25% data studied in Section , the 15% perturbation data set is:

- Dimension $P = 939$

- Training set size: $l = 800$

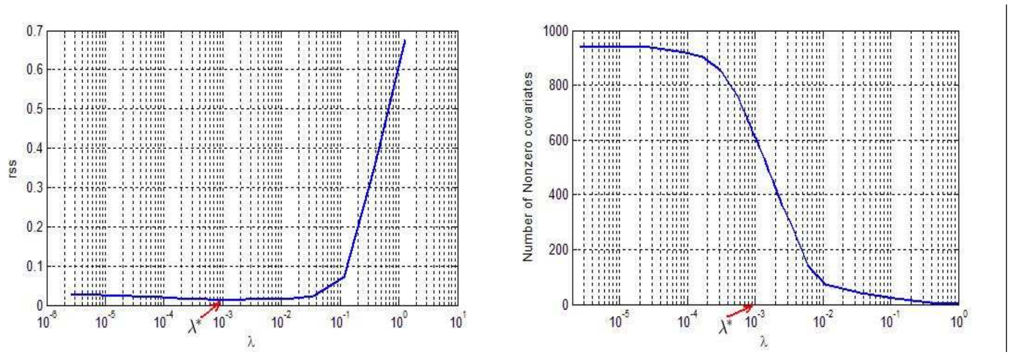
The length of testing set is $l' = 400$. Data are normalized before applying regression techniques. The regression here is exactly similar to that in Section 5.5.

5.5.1 Lasso Regression Analysis, Using 4-fold CV for Parameter Selection

The setup is:

- Use 4-fold of Cross-Validation to select regularization parameter λ.
- $T_1 = 2000$.
- $T_2 = 15000$.

The mean square residual for the constrained fit with constraint λ, and the nonzero weights number vs λ are shown in the Fig. 58. Note that λ^*_{CV} is emphasized as λ^* in the figure.



(a) (b)

Figure 58: Mean Square residual (MSR) and # of selected features vs λ in Section 5.5.1, (a) Mean square residual (MSR) vs λ, (b) Number of selected features vs λ.

MSR is minimum (0.0124) when $\lambda = 9.2039 \times 10^4$ and the result implies that 628 of the total 939 features are with nonzero weights. Note that this number is different from the final number of selected features, since in parameter selection, $T_1 = 2000$ is a relatively small number of iteration for the "shooting algorithm" to converge. The reasons to use such a small T_1 was discussed in Section 5.4.6. Then we use $\lambda^{(1)} = \lambda^*_{CV} = 9.2039 \times 10^4$ to calculate the lasso regression weights for the whole training data set. Denote the estimated weights in this step by $\hat{\beta}^{(1)}$. The superscript here denotes it is the estimate in first-step Lasso. Then we find that the number of nonzero weights is 341. This is true number of selected features using $\lambda^{(1)} = 9.2039 \times 10^4$. We apply the linear model with weights $\hat{\beta}^{(1)}$ to the testing set and get $\{\hat{Y}_j^{(1)},\ j = 1, \cdots, 400\}$. We calculate the prediction error by MSE $= \frac{1}{l'} \sum_{j=1}^{l'} (Y_j - \hat{Y}_j^{(1)})^2$. ($l' = 400$) The result is 0.019694.

Then we perform multi-step Lasso regression, the result is shown in Table 27. We also examine the pattern classification indices. Interestingly there is no classification error for the testing set (shown in Table 28).

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00092039	0.01242441	0.01969405	0.28528630	341
2	0.00016090	0.00736680	0.01461748	0.24578177	53
3	0.00002414	0.00688278	0.01421118	0.24234189	43
4	0.00001053	0.00685143	0.01404045	0.24088175	43
5	0.00001114	0.00685039	0.01402272	0.24072959	43
6	0.00000807	0.00684814	0.01403481	0.24083334	42

Table 27: Multi-step Lasso regression for linear model of 15% disturbance data . 4-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0	0	0	8.000- 8.000	8.000 - 8.000
2	0	0	0	8.000- 8.000	8.000 - 8.000
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
6	0	0	0	8.000- 8.000	8.000 - 8.000

Table 28: Classification indices for applying Multi-step Lasso regression for linear model of 15% disturbance data. 4-fold CV used in selecting regularization parameter, $T_1 = 2000$ and $T_2 = 15000$.

5.5.2 Lasso and Ridge Regression: Comparison Studies

Similar to Section 5.4.3. Suppose we also use 4-fold Cross-Validation to select the regularization parameter in (67). First, the relationship between residual sum of squares and regularization parameter λ is shown in Fig. 59.

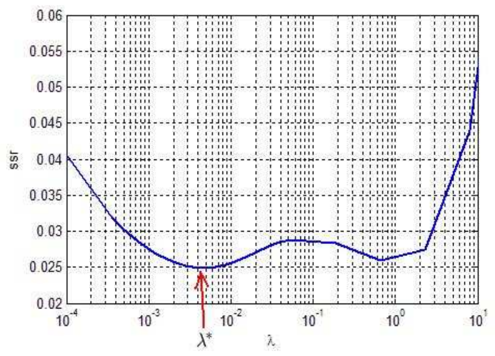


Figure 59: Mean square residual (MSR) vs λ in Cross-Validation in Ridge regression in Section 5.5.2

So the optimal λ for Ridge Regression is 0.0046308. By using this regularization parameter and the whole training set, we calculate the weights using Ridge Regression. Then we use the testing set to evaluate the error. The mean square error of prediction corresponds to Ridge Regression is 0.0249284. The performance for regression as well as pattern classification is shown in Table 29 and Table 30.

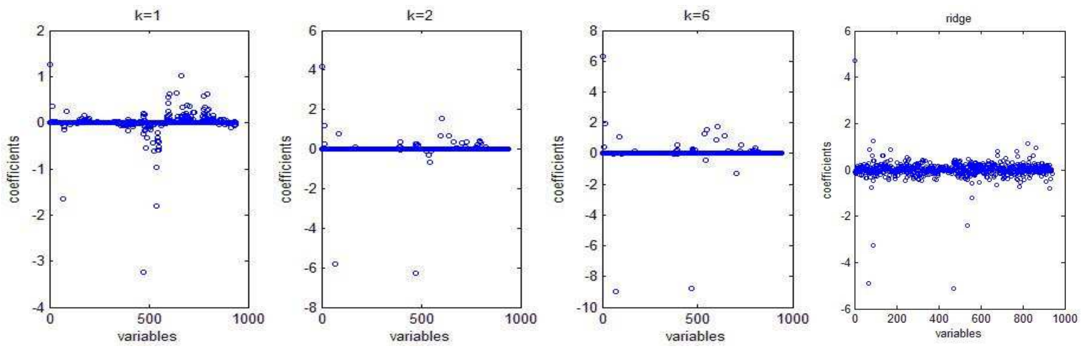
Ridge	λ_{CV}	MSR	MSE	RMSE(cycles)	# F
	0.0046308	0.0248536	0.0249284	0.32096721	939

Table 29: Ridge Regression for 15% disturbance data. 4-fold CV used in selecting regularization parameter.

Ridge	FD	FA	FC	FD range (cycles)	FA range (cycles)
	0	0	0	8.000 - 8.000	8.000 - 8.000

Table 30: Classification indices for applying Ridge Regression. 4-fold CV used in selecting regularization parameter.

The 5-th step Lasso regression shown in Table 27 has a 43.75% smaller mean square error compared with Ridge Regression, or equivalently, a 25.00% smaller root mean square error compared with Ridge Regression. Also Lasso regression lead to the selection of 43 features, compared to the total 939 features, while Ridge Regression leave all features to have non-zero weights. The weights of the features are shown in Fig. 60, Lasso, Adaptive Lasso, 6-step Lasso results are compared with Ridge Regression results.



(a) (b) (c) (d)

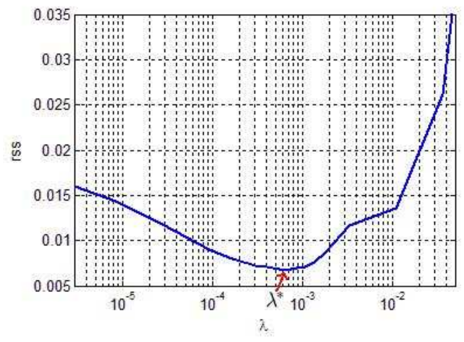
Figure 60: The weights determined by regression. (a) One-step Lasso, (b) Two-step Lasso, (c) Six-step Lasso, (d) Ridge Regression.

5.5.3 Lasso Regression for Quadratic Models

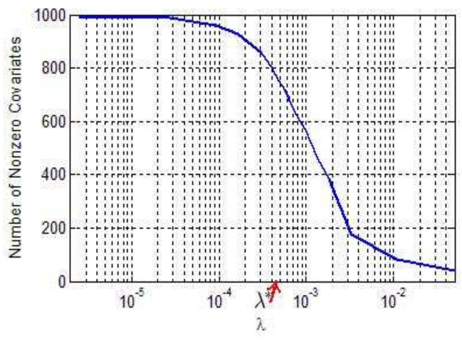
Similar to section 3.2.4, we want to do regression to include quadratic relations of features. In order to simplify computation, we expand the models from the 43 features selected by the 3-rd step Lasso regression. (See Table 27 in Section 5.5.1) The total extended model includes 989 features, in which $43+41+\cdots+1 = 946$ covariates are quadratic terms, and 43 covariates are the original features. Then we apply multi-step Lasso regression, where the parameters are set-up as follows:

- Use 3-fold Cross-Validation to select regularization parameter λ,
- $T_1 = 3000$,
- $T_2 = 15000$.

In the first step Lasso, the relationship between regression sum of squares and λ are shown in Fig. 61, together with the corresponding number of nonzero-weight features vs λ. Minimum MSR corresponds to $\lambda = 0.00062592$. Use this as the regularization parameter in the first step Lasso. The Multi-step Lasso regression results are shown in Table 31. And the classification performance of different steps of Lasso are shown in Table 32. Again, the classification error is zero for the testing set.



(a)



(b)

Figure 61: (a) Mean squre residual (MSR) vs λ, (b) Number of selected features vs λ.

step k	$\lambda^{(k)}$	MSR	MSE	RMSE(cycles)	# F
1	0.00062592	0.00667564	0.01226547	0.22514148	275
2	0.00000625	0.00575407	0.01002906	0.20358396	164
3	0.00006055	0.00537738	0.01062372	0.20953268	114
4	0.00001693	0.00480929	0.01016227	0.20493149	104
5	0.00002793	0.00460699	0.01010217	0.20432467	94
6	0.00002403	0.00456329	0.01010768	0.20438038	92

Table 31: Multi-step Lasso regression for extended model of 15% disturbance data . 3-fold CV used in selecting regularization parameter, $T_1 = 3000$ and $T_2 = 15000$.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
1	0	0	0	8.000- 8.000	8.000 - 8.000
2	0	0	0	8.000- 8.000	8.000 - 8.000
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
6	0	0	0	8.000- 8.000	8.000 - 8.000

Table 32: Classification indices for applying Multi-step Lasso regression for extended model of 15% disturbance data. 3-fold CV used in selecting regularization parameter, $T_1 = 3000$ and $T_2 = 15000$.

5.5.4 Lasso Algorithm and Kernel Ridge Regression: Comparison Studies

Similar to Section 5.4.5, Kernel Ridge Regression parameter q and λ are selected by Cross-Validation. Here we use 4-fold Cross-Validation. Regressions are conducted applying quadratic kernel of different orders. The results are shown in Table 33 and Table 34.

order p	$q_{3CV}^{(p)}$	$\lambda_{3CV}^{(p)}$	MSR	MSE	RMSE(cycles)	# F
2	339820833	542.46909370	0.01730951	0.01619021	0.40619259	939
3	697830585	1.218814e+12	0.01729585	0.01619687	0.40627610	939
4	777365030	1.407465e+21	0.01730467	0.01616100	0.40582600	939
5	388905200	1.018152e-016	0.01963762	0.01841108	0.43315699	939
6	562341325	1.018152e-016	0.01966699	0.01856998	0.43502226	939
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
12	897687132	1.018152e-016	0.01965686	0.01813012	0.42983919	939

Table 33: Kernel Ridge Regression of different orders for 15% disturbance data. 4-fold CV used in selecting regularization parameter.

step k	FD	FA	FC	FD range (cycles)	FA range (cycles)
2	2.26%	1.75%	4.01%	7.188 - 8.000	8.000 - 17.102
3	2.26%	1.75%	4.01%	7.188 - 8.000	8.000 - 17.102
4	2.26%	1.75%	4.01%	7.188 - 8.000	8.000 - 17.102
5	2.26%	1.50%	3.76%	7.188 - 8.000	8.000 - 14.466
6	2.26%	1.75%	4.01%	7.188 - 8.000	8.000 - 17.102
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
12	2.26%	1.50%	3.76%	7.188 - 8.000	8.000 - 14.466

Table 34: Classification indices for applying Kernel Ridge Regression of different orders for 15% disturbance data. 4-fold CV used in selecting regularization parameter.

5.6 Conclusions for Transient Stability Analysis

In this section, we have examined the Lasso algorithm in the context of transient stability analysis. Our results show that Lasso algorithm and its extensions outperform the commonly used techniques utilizing Ridge Regression and Kernel Ridge Regression techniques in terms of the prediction error. For the 25% perturbation data, the properly tuned Lasso regression leads to a 38% smaller MSE error compared with Kernel Ridge Regression. Furthermore due to the adaptive nature of Lasso algorithm, we achieve not only a smaller prediction error bust also a more parsimonious model compared with the solutions employing the l_2 penalty. Depending on the purpose of application, one can choose between the models that lead to more precise prediction, and the ones that have smaller number of feature variables. By selecting loop parameters T_1 and T_2, the balance between accuracy and efficiency can also be selected based on the application purpose of the user.

References

[1] E. Capobianco, “Hammerstein System Represention of Financial Volatility Processes,” *The European Physical Journal B*, vol. 27, pp. 201–211, 2002.

[2] G. Giannakis and E. Serpedin, “A Bibliography on Nonlinear System Identification,” *Signal Processing*, vol. 81, pp. 533–580, 2001.

[3] I. Goethals, K. Pelckmans, J. Suykens, and B. De Moor, “Identification of MIMO Hammerstein Models Using Least Squares Support Vector Machines,” *Automatica*, vol. 41, pp. 1263–1272, 2005.

[4] G. Harnischmacher and W. Marquardt, “A Multi-variate Hammerstein Model for Processes with Input Directionality,” *Journal of Process Control*, vol. 17, pp. 539–550, 2007.

[5] S. Lakshminarayanan, S. Shah, and K. Nandakumar, “Identification of Hammerstein Models Using Multivariate Statistical Tools,” *Chemical Engineering Science*, vol. 50, pp. 3599–3613, 1995.

[6] T. Kara and I. Eker, “Nonlinear Modeling and Identification of a DC Motor for Bidirectional Operation with Real Time Experiments,” *Energy Conversion and Management*, vol. 45, pp. 1087–1106, 2004.

[7] T. Quatieri, D. Reynolds, and G. O’Leary, “Estimation of Handset Nonlinearity with Application to Speaker Recognition,” *IEEE Transactions on Speech and Audio Processing*, vol. 8, pp. 567–584, 2000.

[8] M. Gevers, “A Personal View of the Development of System Identification,” *IEEE Control Systems Magazine*, vol. 26, pp. 93–105, 2006.

[9] M. Deistler, “System Identification and Time Series Analysis: Past, Present, and Future,” *Stochastic Theory and Control*, pp. 97–109, 2002.

[10] K. Aström and T. Bohlin, “Numerical Identification of Linear Dynamic Systems from Normal Operating Records,” in *Theory of Self-adaptive Control Systems*, 1966, pp. 96–111.

[11] L. Ljung, *System Identification: Theory for the User*. Prentice-Hall Englewood Cliffs, NJ, 1987, vol. 11.

[12] W. Greblicki and M. Pawlak, *Nonparametric System Identification*. Cambridge University Press, 2008.

[13] M. Campi and E. Weyer, “Finite Sample Properties of System Identification Methods,” *IEEE Transactions on Automatic Control*, vol. 47, pp. 1329–1334, 2002.

[14] F. Giri and E. Bai, *Block-Oriented Nonlinear System Identification*. Springer-Verlag, 2010.

[15] D. Westwick and R. Kearney, *Identification of Nonlinear Physiological Systems*. Wiley-IEEE Press, 2003.

[16] E. Eskinat, S. Johnson, and W. Luyben, “Use of Hammerstein Models in Identification of Nonlinear Systems,” *AIChE Journal*, vol. 37, pp. 255–268, 1991.

[17] K. Hunt, M. Munih, N. Donaldson, and F. Barr, “Investigation of the Hammerstein Hypothesis in the Modeling of Electrically Stimulated Muscle,” *IEEE Transactions on Biomedical Engineering*, vol. 45, pp. 998–1009, 1998.

[18] S. Kukreja, R. Kearney, and H. Galiana, “A Least-Squares Parameter Estimation Algorithm for Switched Hammerstein Systems with Applications to the VOR,” *IEEE Transactions on Biomedical Engineering*, vol. 52, pp. 431–444, 2005.

[19] J. Turunen, J. Tanttu, and P. Loula, “Hammerstein Model for Speech Coding,” *EURASIP Journal on Applied Signal Processing*, pp. 1238–1249, 2003.

[20] A. Hammerstein, “Nichtlineare Integralgleichungen Nebst Anwendungen,” *Acta Mathematica*, vol. 54, pp. 117–176, 1930.

[21] J. Fan and Q. Yao, *Nonlinear Time Series: Nonparametric and Parametric Methods*. Springer-Verlag, 2005.

[22] M. Pawlak, “Identification of Block-oriented Systems: Nonparametric and Semiparametric Inference,” *Block-oriented Nonlinear System Identification*, pp. 127–146, 2010.

[23] M. Pawlak, Z. Hasiewicz, and P. Wachel, “On Nonparametric Identification of Wiener Systems,” *IEEE Transactions on Signal Processing*, vol. 55, pp. 482–492, 2007.

[24] W. Härdle, *Smoothing Techniques: with Implementation in S*. Springer-Verlag, 1991.

[25] S. Mallat, *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

[26] G. Szegö, *Orthogonal Polynomials*. American Mathematical Society, 1939.

[27] L. Wasserman, *All of Nonparametric Statistics*. Springer Texts in Statistics, Springer-Verlag, 2006.

[28] D. Donoho and J. Johnstone, “Ideal Spatial Adaptation by Wavelet Shrinkage,” *Biometrika*, vol. 81, pp. 425–455, 1994.

[29] J. Neyman, “‘Smooth’ Test for Goodness of Fit.” *Skandinavisk Aktuarietidskrift*, vol. 20, pp. 149–199, 1937.

[30] J. Hart, *Nonparametric Smoothing and Lack-of-Fit Tests*. Springer-Verlag, 1997.

[31] T. Ledwina, “Data-Driven Version of Neyman’s Smooth Test of Fit,” *Journal of the American Statistical Association*, vol. 89, pp. 1000–1005, 1994.

[32] S. Yakowitz, J. Krimmel, and F. Szidarovszky, “Weighted Monte Carlo Integration,” *SIAM Journal on Numerical Analysis*, vol. 15, pp. 1289–1300, 1978.

[33] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1993.

[34] B. Jayasekara, *Determination of Transient Stability Boundary in Functional Form with Applications in Optimal Power Flow and Security Control*. University of Manitoba Ph.D Thesis, 2006.

[35] R. Tibshirani, “Regression Shrinkage and Selection via the Lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, pp. 267–288, 1996.

[36] B. Jayasekara and U. Annakkage, "Derivation of an Accurate Polynomial Representation of the Transient Stability Boundary," *IEEE Transactions on Power Systems*, vol. 21, pp. 1856–1863, 2006.

[37] L. Moulin, A. Da Silva, M. El-Sharkawi, R. Marks, *et al.*, "Support vector machines for transient stability analysis of large-scale power systems," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 818–825, 2004.

[38] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani, "Pathwise coordinate optimization," *The Annals of Applied Statistics*, vol. 1, no. 2, pp. 302–332, 2007.

[39] H. Zou, "The adaptive Lasso and Its Oracle Properties," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.