A Novel Algorithm for Nonlinear Dynamical System Filtering

McGill University, Montreal

Department of Electrical and Computer Engineering

Author:

Jiangning Luo

Supervisor:

Prof. Hannah Michalska

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of MSc Electrical Engineering

December 15, 2020

Abstract

A novel algorithm for estimating nonlinear dynamical systems' true output from noisy measurements is presented in this thesis. The thesis begins with a presentation of an algorithm to estimate the output of linear dynamical systems that outperform the linear Kalman filter [15]. Then presented the development of a Kalman like filter for nonlinear dynamical systems which is embedding the measurements into reproducing kernel Hilbert space (RKHS) [35]. Our novel algorithm is inspired by the Kalman like filter for nonlinear dynamical systems (KKF-CEO), which is adapting the algorithm that outperforms the linear Kalman filter for nonlinear dynamical systems by embedding the measurements into RKHS. Our novel algorithm is then applied to estimate the true outputs of a linear system and several nonlinear systems. Our algorithm is shown to be more accurate than KKF-CEO in most of the experiments, and our algorithm runs much faster than KKF-CEO. Additionally, in contrast to KKF-CEO, our novel algorithm does not need parameter tuning to achieve better performance.

Abrégé

Un nouvel algorithme pour estimer la sortie réelle de systèmes dynamiques non linéaires à partir de mesures bruyantes est présenté dans cette thèse. La thèse commence par un algorithme qui estime la vraie sortie de systèmes dynamiques linéaires et surpasse le filtre de Kalman linéaire. Puis passe à l'introduction d'un algorithme qui adapte le filtre de Kalman linéaire pour les systèmes dynamiques non linéaires en incorporant les mesures dans l'espace de Hilbert du noyau de reproduction (RKHS). Le nouvel algorithme est inspiré du filtre de Kalman pour les systèmes dynamiques non linéaires (KKF-CEO), qui adapte l'algorithme qui surpasse le filtre de Kalman linéaire pour les systèmes dynamiques non linéaires en intégrant les mesures dans RKHS. Ensuite, notre nouvel algorithme est appliqué pour estimer les véritables sorties d'un système linéaire et de plusieurs systèmes non linéaires. Notre algorithme a une meilleure précision que KKF-CEO dans la plupart des expériences et notre algorithme s'exécute beaucoup plus rapidement que KKF-CEO. De plus, aucun paramètre ne doit être réglé dans notre algorithme tandis que deux paramètres doivent être ajustés en fonction de la connaissance préalable du bruit afin d'obtenir de bonnes performances dans KKF-CEO.

Acknowledgements

I am grateful to several individuals who helped me constantly throughout my two years of academic life at McGill.

Firstly, thank my supervisor Hannah Michalska for the generous help, she guided me patiently throughout my research and helped me with my thesis.

Secondly, thank my family for giving me support both financially and emotionally.

And finally, thank my friend Huiyuan Yang who encouraged me and given me a lot of help during my research.

Preface

In chapter two, I introduced the classic Kalman filter and RKHS in detail, and then reviewed some popular algorithms used on nonlinear dynamical systems. In chapter three, I summarized the master thesis by Anju John et al. In chapter four, I summarized the paper by Zhu et al. Chapter five is the derivation of my novel algorithm. Chapter six consists of the results of my experiments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A nonlinear dynamical system is a system that does not obey the superposition principle (see Definition 1.0.1). Nonlinear dynamical systems proliferate as models in fields of engineering, mathematics, physics, and biology. Nearly all of the real-world dynamical systems are nonlinear dynamical systems. Nonlinear dynamical systems may appear unpredictable and chaotic, making them harder to study and control than linear dynamical systems.

Definition 1.0.1. Superposition Principle: [13] The superposition principle is defined by the following two properties:

$$F(u_1 + u_2) = F(u_1) + F(u_2) \tag{1.1}$$

$$F(\alpha u_1) = \alpha F(u_1)\ \forall u_1, u_2,\ \forall \alpha \in \mathbb{R} \tag{1.2}$$

where u_1, u_2 are system inputs and F represents the system input-output mapping: $y = F(u)$. Also, α is any constant parameter.

In this thesis, we focus on the nonlinear dynamical systems. The model we are using in this thesis can be described as Figure 1.1. Assume we are measuring the outputs generated by the nonlinear system in the time interval $[a, b]$ sampled with time increment Δt, $\mathbf{x}_i \in \mathbb{R}^n$ is the output generated by the system at time $a + i\Delta t$ which is usually not

observable, $\mathbf{y}_i$ is the measurement of $\mathbf{x}_i$ and $\mathbf{v}_i$ is the independent noise added to $\mathbf{x}_i$. Our objective is to estimate the output sequence $\{\mathbf{x}_i\}$ using the noisy measurements $\{\mathbf{y}_i\}$ without knowing the underlying model for the nonlinear system.

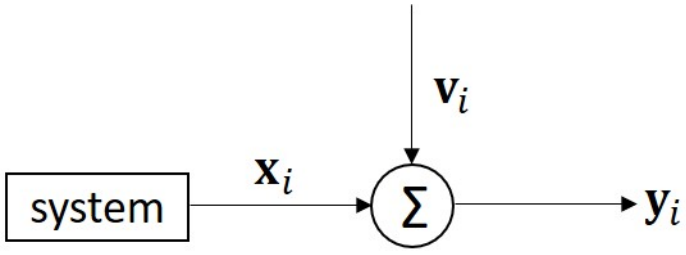

Figure 1.1: Observed time series model.

1.1 Thesis Objectives and Organization

This Master thesis's main objective is to develop a new algorithm for output estimation from noisy output data. To accomplish this, an algorithm developed as an improvement of Kalman filter [8] and an algorithm based on the concept of Kalman filter [35] are studied. The bases of the second algorithm: the classical Kalman filter and reproducing kernel Hilbert space (RKHS) are also studied.

Chapter 2 introduces the classical Kalman filter and RKHS in detail, then several popular algorithms used on nonlinear dynamical systems are reviewed.

Chapter 3 focuses on the derivation and understanding of the double sided kernel (KDS) for single input single output (SISO) linear systems.

Chapter 4 introduces the derivation and the procedures of a Kalman like filter for nonlinear dynamical system (KKF-CEO), which is embedding the measurements into reproducing kernel Hilbert space (RKHS).

Chapter 5 presents our new algorithm's derivation, which is adapting the double sided kernel for nonlinear dynamical systems.

Chapter 6 compares the performances of KDS, KKF-CEO and our novel algorithm for lin-

ear systems, followed by the comparison between the performances of KKF-CEO and our novel algorithm for nonlinear systems.

Chapter 7 concludes the thesis and discusses the future improvements of our novel algorithm.

Chapter 2

Background

This chapter summarizes the principles underlying the classical Kalman filter and introduces a special kind of Hilbert space: the Reproducing Kernel Hilbert space (RKHS). Both are instrumental in the development of the novel adaptive Kalman filter algorithm presented here. Then, some popular algorithms used on nonlinear systems related to the Kalman filter and RKHS are reviewed in Section 2.3.

2.1 Kalman Filter [10] [35]

Kalman filter is an algorithm proposed by R. E. Kalman [19], which is widely used for state estimation in linear dynamical systems whose mathematical model is known. In a Kalman filter, the system state is estimated recursively. It is computed by combining the previous state estimate and the new observed measurement. Therefore only the previous state estimate needs to be stored. Kalman filter is useful in various applications, and it is computationally simple.

2.1.1 Problem Definition

As noted above, the Kalman filter serves as an estimator of the state of linear dynamical systems described by Figure 2.1 [10]. The state vector $\mathbf{x}_i$ is the state at time instance

i, which is describing the system's dynamical behavior. State vectors $\{\mathbf{x}_i\}$ are seldom known or measured, hence is the need for its estimation from the output measurement data. A set of measurements $\{\mathbf{y}_i\}$ are used to estimate the unknown state vectors.

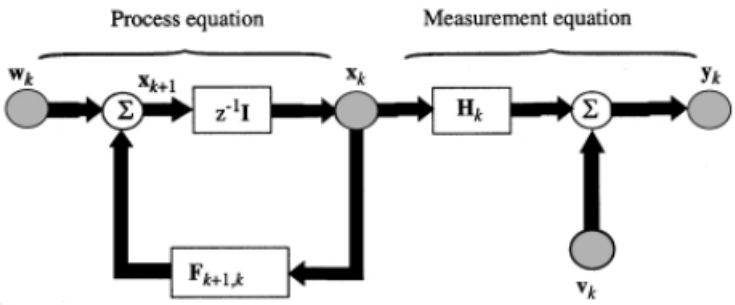

Figure 2.1: Block diagram representing a linear, discrete-time dynamical system. [10]

Mathematically, the block diagram in Figure 2.1 can be expressed by the following equations:

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{F}_{i+1,i}\mathbf{x}_i + \mathbf{w}_i \\ \mathbf{y}_i &= \mathbf{H}_i\mathbf{x}_i + \mathbf{v}_i \end{aligned} \tag{2.1}$$

The first equation is called process equation, where $\mathbf{F}_{i+1,i}$ is called the transition matrix, which takes the state $\mathbf{x}_i$ from time instance i to time instance $i+1$. With reference to Figure 2.1, $\mathbf{w}_i$ denotes the process noise, which is assumed to be white Gaussian with zero mean and covariance matrix $E[\mathbf{w}_n\mathbf{w}_i^T]$,

$$E[\mathbf{w}_n\mathbf{w}_i^T] = \begin{cases} \mathbf{Q}_i \text{ for } n = i \\ 0 \text{ for } n \neq i \end{cases} \tag{2.2}$$

The second equation is called the measurement equation, where $\mathbf{y}_i$ is the measurement at time instance i, $\mathbf{H}_i$ is called the measurement matrix. The measurement noise $\mathbf{v}_i$ is also

assumed to be white Gaussian with zero mean and covariance matrix $E[\mathbf{v}_n\mathbf{v}_i^T]$,

$$E[\mathbf{v}_n\mathbf{v}_i^T] = \begin{cases} \mathbf{R}_i \text{ for } n = i \\ 0 \text{ for } n \neq i \end{cases} \tag{2.3}$$

It is also assumed that the measurement noise $\mathbf{v}_i$ is independent of the process noise $\mathbf{w}_i$. There are three types of Kalman algorithms: filtering, prediction and smoothing. The Kalman filtering problem is defined as follows: [10]

Definition 2.1.1. Kalman filtering problem: [10] Use the entire measurements consist of vectors $\mathbf{y}_1$, $\mathbf{y}_2$,..., $\mathbf{y}_k$, $k \geq 1$ to find the minimum mean-square error estimate of the state $\mathbf{x}_i$. When $i = k$, the problem is called filtering.

2.1.2 Optimality of the Kalman Filter

For simplicity, consider the following equation:

$$\mathbf{y}_i = \mathbf{x}_i + \mathbf{v}_i \tag{2.4}$$

where $\mathbf{x}_i$ is an unknown signal and $\mathbf{v}_i$ is the additive noise. The estimate of unknown $\mathbf{x}_i$ given the measurements $\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_i$ is denoted as $\hat{\mathbf{x}}_i$. To quantify the difference between the estimate $\hat{\mathbf{x}}_i$ and the unknown $\mathbf{x}_i$, a loss function is needed. The loss function should satisfy the following two requirements:

1. It must be non negative.

2. It must be a non decreasing function with respect to the estimation error $\tilde{\mathbf{x}}_i$ which is defined by

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i - \hat{\mathbf{x}}_i \tag{2.5}$$

The mean square error satisfies these two requirements and is defined by

$$J_i = E(||\mathbf{x}_i - \hat{\mathbf{x}}_i||^2) = E(||\tilde{\mathbf{x}}||_i^2) \tag{2.6}$$

where E denotes the expectation operator.
The following two theorems taken from [19] and [30] are useful for calculating the optimal value of the estimate $\hat{\mathbf{x}}_i$:

Theorem 2.1.1. Conditional mean estimator: [19] If $\{\mathbf{x}_i\}$ and $\{\mathbf{y}_i\}$ are jointly Gaussian stochastic processes, the conditional mean estimator is the optimal value of the estimate $\hat{\mathbf{x}}_i$ which minimizes the mean-square error J_i:

$$\hat{\mathbf{x}}_i = E[\mathbf{x}_i|\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_i] \tag{2.7}$$

Theorem 2.1.2. Principle of orthogonality: [30] Consider the stochastic processes $\{\mathbf{x}_i\}$ and $\{\mathbf{y}_i\}$ with zero means, assume that

$$E[\mathbf{x}_i] = E[\mathbf{y}_i] = 0 \text{ for all } k \tag{2.8}$$

If the stochastic processes $\{\mathbf{x}_i\}$ and $\{\mathbf{y}_i\}$ are jointly Gaussian, or if we restrict the optimal estimate $\hat{\mathbf{x}}_i$ to be a linear function of the measurements and the loss function is mean-square error. Then the optimal estimate $\hat{\mathbf{x}}_i$ given the measurements $\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_i$ is the orthogonal projection of $\mathbf{x}_i$ on the space spanned by the measurements up to time instance i.

The Kalman filter can be derived by using these two theorems.

2.1.3 Kalman Filter [19]

Consider the linear dynamical system described by (2.1). At time instance i, the information in the new measurement $\mathbf{y}_i$ is required to update the estimate of the unknown

state $\mathbf{x}_i$. The a priori estimate of the unknown state which is available at time instance i is denoted as $\hat{\mathbf{x}}_i^-$. Because the a posteriori estimate $\hat{\mathbf{x}}_i$ is restricted to be a linear function of the measurements up to time instance i, it can be expressed as a linear combination of a priori estimate and the new measurement:

$$\hat{\mathbf{x}}_i = \mathbf{G}_i^{(1)}\hat{\mathbf{x}}_i^- + \mathbf{G}_i\mathbf{y}_i \tag{2.9}$$

The matrix factors $\mathbf{G}_i^{(1)}$ and $\mathbf{G}_i$ need to be determined. To determine the two matrix factors, Theorem 2.1.2 will be used. The state-error vector is:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i - \hat{\mathbf{x}}_i \tag{2.10}$$

By Theorem 2.1.2, we can get

$$E[\tilde{\mathbf{x}}_i\mathbf{y}_i^T] = \mathbf{0} \text{ for } i = 1, 2, ..., k-1 \tag{2.11}$$

From (2.1), (2.9), (2.10) and (2.11), we can get

$$E[(\mathbf{x}_i - \mathbf{G}_i^{(1)}\hat{\mathbf{x}}_i^- - \mathbf{G}_i\mathbf{H}_i\mathbf{x}_i - \mathbf{G}_i\mathbf{w}_i)\mathbf{y}_i^T] = \mathbf{0} \text{ for } i = 1, 2, ..., k-1 \tag{2.12}$$

Because the process noise $\mathbf{w}_i$ and the measurement $\mathbf{y}_i$ are independent, we have

$$E[\mathbf{w}_i\mathbf{y}_i^T] = \mathbf{0} \tag{2.13}$$

Then, (2.12) can be rewritten as

$$E[(\mathbf{I} - \mathbf{G}_i\mathbf{H}_i - \mathbf{G}_i^{(1)})\mathbf{x}_i\mathbf{y}_i^T + \mathbf{G}_i^{(1)}(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)\mathbf{y}_i^T] = \mathbf{0} \tag{2.14}$$

Where $\mathbf{I}$ is identity matrix. From Theorem 2.1.2, we have

$$E[(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)\mathbf{y}_i^T] = 0 \tag{2.15}$$

Then, (2.12) can be further simplified to

$$(\mathbf{I} - \mathbf{G}_i\mathbf{H}_i - \mathbf{G}_i^{(1)})E[\mathbf{x}_i\mathbf{y}_i^T] = \mathbf{0} \text{ for } i = 1, 2, ..., k-1 \tag{2.16}$$

If (2.16) satisfies for arbitrary states $\mathbf{x}_i$ and measurements $\mathbf{y}_i$, the matrix factors $\mathbf{G}_i^{(1)}$ and $\mathbf{G}_i$ need to satisfy the following relation:

$$\mathbf{I} - \mathbf{G}_i\mathbf{H}_i - \mathbf{G}_i^{(1)} = \mathbf{0} \tag{2.17}$$

Hence, $\mathbf{G}_i^{(1)}$ is defined as

$$\mathbf{G}_i^{(1)} = \mathbf{I} - \mathbf{G}_i\mathbf{H}_i \tag{2.18}$$

Substituting (2.18) into (2.9), the a posteriori estimate of the state $\mathbf{x}_i$ at time instance i is

$$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_i^- + \mathbf{G}_i(\mathbf{y}_i - \mathbf{H}_i\hat{\mathbf{x}}_i^-) \tag{2.19}$$

The matrix $\mathbf{G}_i$ is called the Kalman gain matrix.

The remaining problem is to derive the Kalman gain matrix $\mathbf{G}_i$. From Theorem 2.1.2, we have

$$E[(\mathbf{x}_i - \hat{\mathbf{x}}_i)\mathbf{y}_i^T] = \mathbf{0} \tag{2.20}$$

hence also

$$E[(\mathbf{x}_i - \hat{\mathbf{x}}_i)\hat{\mathbf{y}}_i^T] = \mathbf{0} \tag{2.21}$$

Here $\hat{\mathbf{y}}_i^T$ denotes the estimation of $\mathbf{y}_i$ given the previous measurements $\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{i-1}$. The innovation process is defined by

$$\tilde{\mathbf{y}}_i = \mathbf{y}_i - \hat{\mathbf{y}}_i \tag{2.22}$$

The innovation process is a measure of how much new information contains in $\mathbf{y}_i$. It can also be expressed as

$$\begin{aligned}\tilde{\mathbf{y}}_i &= \mathbf{y}_i - \mathbf{H}_i\hat{\mathbf{x}}_i^- \\ &= \mathbf{H}_i\mathbf{x}_i + \mathbf{v}_i - \mathbf{H}_i\hat{\mathbf{x}}_i^- \\ &= \mathbf{H}_i\tilde{\mathbf{x}}_i^- + \mathbf{v}_i\end{aligned} \tag{2.23}$$

From (2.20), (2.21) and (2.22), we have

$$E[(\mathbf{x}_i - \hat{\mathbf{x}}_i)\tilde{\mathbf{y}}_i^T] = \mathbf{0} \tag{2.24}$$

From (2.1) and (2.19), the state-error vector $\mathbf{x}_i - \hat{\mathbf{x}}_i$ can be expressed as

$$\begin{aligned}\mathbf{x}_i - \hat{\mathbf{x}}_i &= \tilde{\mathbf{x}}_i^- - \mathbf{G}_i(\mathbf{H}_i\tilde{\mathbf{x}}_i^- + \mathbf{v}_i) \\ &= (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\tilde{\mathbf{x}}_i^- - \mathbf{G}_i\mathbf{v}_i\end{aligned} \tag{2.25}$$

Bring (2.23) and (2.25) into (2.24), we have

$$E[\{(\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\tilde{\mathbf{x}}_i^- - \mathbf{G}_i\mathbf{v}_i\}(\mathbf{H}_i\tilde{\mathbf{x}}_i^- + \mathbf{v}_i)] = \mathbf{0} \tag{2.26}$$

The measurement noise $\mathbf{v}_i$ is independent of the state $\mathbf{x}_i$ and therefore independent of the error $\tilde{\mathbf{x}}_i^-$, (2.26) can be reduced to

$$(\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)E[\tilde{\mathbf{x}}_i^-\tilde{\mathbf{x}}_i^{T-}]\mathbf{H}_i^T - \mathbf{G}_iE[\mathbf{v}_i\mathbf{v}_i^T] = \mathbf{0} \tag{2.27}$$

The a priori covariance matrix $\mathbf{P}_i^-$ is defined as

$$\begin{aligned}\mathbf{P}_i^- &= E[(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)^T] \\ &= E[\tilde{\mathbf{x}}_i^-\tilde{\mathbf{x}}_i^{T-}]\end{aligned} \tag{2.28}$$

Invoking (2.3) and (2.28), (2.27) can be rewritten as

$$(\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^-\mathbf{H}_i^T - \mathbf{G}_i\mathbf{R}_i = \mathbf{0} \tag{2.29}$$

Finally, we have the formula of the Kalman gain matrix $\mathbf{G}_i$

$$\mathbf{G}_i = \mathbf{P}_i^-\mathbf{H}_i^T[\mathbf{H}_i\mathbf{P}_i^-\mathbf{H}_i^T + \mathbf{R}_i]^{-1} \tag{2.30}$$

where $[\cdot]^{-1}$ denotes the matrix inverse.

Next, consider the recursive process of updating the covariance matrix of the estimation error, which is called error covariance propagation. Covariance propagation involves two steps of computation: [10]

1. (2.28) defines the a priori covariance matrix $\mathbf{P}_i^-$ at time instance i. Given the a priori covariance matrix $\mathbf{P}_i^-$, compute the a posteriori covariance matrix $\mathbf{P}_i$ at time instance i, which is defined by

$$\mathbf{P}_i = E[\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T] = E[(\mathbf{x}_i - \hat{\mathbf{x}}_i)(\mathbf{x}_i - \hat{\mathbf{x}}_i)^T] \tag{2.31}$$

2. Given the a posteriori covariance matrix $\mathbf{P}_{i-1}$ at time $i-1$, compute the updated a priori covariance matrix $\mathbf{P}_i^-$ at time instance i.

In stage 1, substitute (2.25) into (2.31). Note that the measurement noise $\mathbf{v}_i$ is independent to the a priori estimate error $\tilde{\mathbf{x}}_i^-$. Then we have

$$\begin{aligned} P_i &= (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)E[\tilde{\mathbf{x}}_i^-\tilde{\mathbf{x}}_i^{T-}](\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)^T + \mathbf{G}_iE[\mathbf{v}_i\mathbf{v}_i^T]\mathbf{G}_i^T \\ &= (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^-(\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)^T + \mathbf{G}_i\mathbf{R}_i\mathbf{G}_i^T \end{aligned} \tag{2.32}$$

Expand (2.32) and bring (2.29) into it, (2.32) can be simplified to

$$\mathbf{P}_i = (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^- - (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^-\mathbf{H}_i^T\mathbf{G}_i^T + \mathbf{G}_i\mathbf{R}_i\mathbf{G}_i^T \tag{2.33}$$

$$= (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^- - \mathbf{G}_i\mathbf{R}_i\mathbf{G}_i^T + \mathbf{G}_i\mathbf{R}_i\mathbf{G}_i^T \tag{2.34}$$

$$= (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^- \tag{2.35}$$

In the second step, firstly consider the formula of a priori estimate of state $\hat{\mathbf{x}}_i^-$ at time instance i in terms of the a posteriori estimate $\hat{\mathbf{x}}_{i-1}$ at time $i-1$:

$$\hat{\mathbf{x}}_i^- = \mathbf{F}_{i,i-1}\hat{\mathbf{x}}_{i-1} \tag{2.36}$$

Then, substitute (2.1) and (2.36) into the a priori estimate error equation, we have

$$\begin{aligned}
\tilde{\mathbf{x}}_i^- &= \mathbf{x}_i - \hat{\mathbf{x}}_i^- \\
&= (\mathbf{F}_{i,i-1}\mathbf{x}_{i-1} + \mathbf{w}_{i-1}) - (\mathbf{F}_{i,i-1}\hat{\mathbf{x}}_{i-1}) \\
&= \mathbf{F}_{i,i-1}(\mathbf{x}_{i-1} - \hat{\mathbf{x}}_{i-1}) + \mathbf{w}_{i-1} \\
&= \mathbf{F}_{i,i-1}\tilde{\mathbf{x}}_{i-1} + \mathbf{w}_{i-1}
\end{aligned} \tag{2.37}$$

Substitute (2.37) into (2.28). Note that the process noise $\mathbf{w}_i$ is independent of $\tilde{x}_{i-1}$, we have

$$\begin{aligned}
\mathbf{P}_i^- &= \mathbf{F}_{i,i-1}E[\tilde{\mathbf{x}}_{i-1}\tilde{\mathbf{x}}_{i-1}^T]\mathbf{F}_{i,i-1}^T + E[\mathbf{w}_{i-1}\mathbf{w}_{i-1}^T] \\
&= \mathbf{F}_{i,i-1}\mathbf{P}_{i-1}\mathbf{F}_{i,i-1}^T + \mathbf{Q}_{i-1}
\end{aligned} \tag{2.38}$$

Which is the updated a priori covariance matrix $\mathbf{P}_i^-$ in terms of the the a posteriori covariance matrix $\mathbf{P}_{i-1}$ at time $i-1$.
The recursive steps of the Kalman filter are summarized in Appendix.

2.2 Reproducing Kernel Hilbert Space (RKHS) [24]

Suppose we have a feature space $\mathcal{X}$, many classical learning algorithms such as perceptron [26], support vector machine (SVM) [4] and principal component analysis (PCA) [25]

employ feature vectors $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ only through an inner product $\langle \mathbf{x}, \mathbf{x}' \rangle$, which is basically a similarity measure between $\mathbf{x}$ and $\mathbf{x}'$. However, the class of linear functions induced by this inner product may be too restrictive for many real-world problems. Therefore, we have kernel methods that aim to build more flexible and powerful learning algorithms by replacing $\langle \mathbf{x}, \mathbf{x}' \rangle$ with some other possibly non-linear similarity measures.

The most natural extention of $\langle \mathbf{x}, \mathbf{x}' \rangle$ is to explicitly apply a non-linear transformation φ:

$$\begin{aligned} \varphi : \mathcal{X} &\to \mathcal{F} \\ \mathbf{x} &\to \varphi(\mathbf{x}) \end{aligned} \tag{2.39}$$

into a high-dimensional feature space $\mathcal{F}$ and subsequently evaluate the inner product in $\mathcal{F}$, i.e.:

$$k(\mathbf{x}, \mathbf{x}') := \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{F}} \tag{2.40}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ denotes the inner product in $\mathcal{F}$. We refer to φ and k as feature map and kernel function respectively. Likewise, we can interpret $k(\mathbf{x}, \mathbf{x}')$ as a non-linear similarity measure between $\mathbf{x}$ and $\mathbf{x}'$. The algorithms that depend on the data sets only through the inner product $\langle \mathbf{x}, \mathbf{x}' \rangle$ can be extended to non-linear by simply substituting $\langle \mathbf{x}, \mathbf{x}' \rangle$ with $\langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{F}}$. Note that the learning algorithm remains the same, we only changes the space in which these algorithm operate. As (2.39) is non-linear, a linear algorithm in $\mathcal{F}$ corresponds to the non-linear counterpart in the original space $\mathcal{X}$.

Evaluating $k(\mathbf{x}, \mathbf{x}')$ as above requires two steps: i) Construct the feature maps $\varphi(\mathbf{x})$ and $\varphi(\mathbf{x})'$ explicitly, ii) Evaluate $k(\mathbf{x}, \mathbf{x}') := \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{F}}$. These two steps can be computationally expensive if $\varphi(\mathbf{x})$ lives in a high-dimensional space. However, $\langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{F}}$ can be evaluated by an alternative way without resorting to constructing $\varphi(\mathbf{x})$ explicitly if all we need is an inner product $\langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{F}}$. That is evaluating $k(\mathbf{x}, \mathbf{x}')$ directly. This is an essential aspect of kernel methods, which often referred to as the kernel trick in machine learning.

When the kernel k is positive definite, there always exists $\varphi : \mathcal{X} \to \mathcal{F}$ for which $k(\mathbf{x}, \mathbf{x}') =$

$\langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{F}}$, then the kernel trick will be possible [24]. The definition of positive definite kernel is:

Definition 2.2.1. Positive definite kernel: [24] A function $k : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$ is a positive definite kernel if it is symmetric, i.e., $k(\mathbf{x}, \mathbf{y}) = k(\mathbf{y}, \mathbf{x})$, and the Gram matrix is positive definite:

$$\mathbf{c}^T \mathbf{K} \mathbf{c} = \sum_{i,j=1}^{n} c_i c_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \tag{2.41}$$

for any $n \in \mathbf{N}$, any choice of $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathcal{X}$ and any $c_1, ..., c_n \in \mathbf{R}$. It is said to be strictly positive definite if the equality in (2.41) implies $c_1 = c_2 ... = c_n = 0$.

A positive definite kernel defines a space of functions from $\mathcal{X}$ to $\mathbf{R}$ called reproducing kernel Hilbert space (RKHS) and denoted by $\mathcal{H}$. Hence the kernel k is also called a reproducing kernel. Whenever we use the kernel k, the feature space $\mathcal{F}$ is essentially the RKHS $\mathcal{H}$ associated with the kernel k and we often think of a canonical feature map

$$\begin{aligned} k : \mathcal{X} &\to \mathcal{H} \subset \mathbf{R}^{\mathcal{X}} \\ \mathbf{x} &\to k(\mathbf{x}, \cdot) \end{aligned} \tag{2.42}$$

where $\mathbf{R}^{\mathcal{X}}$ denotes the vector space of functions from $\mathcal{X}$ to $\mathbf{R}$. All RKHS satisfy the Moore-Aronszajn Theorem:

Theorem 2.2.1. Moore-Aronszajn Theorem [2]:
Any non-negative definite kernel function $k(x, y)$, uniquely defines a corresponding Hilbert space $\mathcal{H}$ which consists of all functions f defined on a common domain $\mathcal{X} \subset \mathbb{R}$ which satisfy the following properties

1. $\forall x \in \mathcal{X}$, the function $k(\mathbf{x}, \cdot) : \mathbf{y} \to k(\mathbf{x}, \mathbf{y})$ is an element of $\mathcal{H}$
2. $\forall x \in \mathcal{X}, \forall f \in \mathcal{H}$, $f(x) = \langle f, k(\mathbf{x}, \cdot) \rangle_{\mathcal{H}}$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product in $\mathcal{H}$. The second property mentioned above is also known as the reproducing property of kernel k.

For convenience, $k(\mathbf{x}, \cdot)$ above can be denoted as $k_{\mathbf{x}}$. Similarly, $k(\mathbf{y}, \cdot)$ can be denoted as $k_{\mathbf{y}}$. Then, by reproducing property, we have

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= k_{\mathbf{y}}(\mathbf{x}) \\ &= \langle k_{\mathbf{x}}, k_{\mathbf{y}} \rangle_{\mathcal{H}} \\ &= \langle \varphi(\mathbf{x}), \varphi(\mathbf{y}) \rangle_{\mathcal{H}} \end{aligned} \tag{2.43}$$

Therefore, the feature map $\varphi(\mathbf{x})$ can be viewed as $k(\mathbf{x}, \cdot)$ alternatively. Then we have

$$\forall x \in \mathcal{X}, \forall f \in \mathcal{H}, f(x) = \langle f, \varphi(\mathbf{x}) \rangle_{\mathcal{H}} \tag{2.44}$$

2.3 Related Works

2.3.1 Extended Kalman Filter

Extended Kalman filter (EKF) is an adapted version of Kalman filter that is used on non-linear state-space systems. The basic idea of EKF is to linearize the nonlinear state-space model

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{f}_i(\mathbf{x}_i, \mathbf{u}_i) + \mathbf{n}_i \\ \mathbf{y}_i &= \mathbf{h}_i(\mathbf{x}_i.\mathbf{u}_i) + \mathbf{v}_i \end{aligned} \tag{2.45}$$

around the most recent state estimation, either $\hat{\mathbf{x}}_i$ or $\hat{\mathbf{x}}_i^-$. The linearized state-transaction matrices are

$$\begin{aligned} \mathbf{F}_i &= \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}_{i-1}} \\ \mathbf{H}_i &= \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}_i^-} \end{aligned}$$

Once the linearization is done, the standard Kalman filter can be applied to the linearized system. The noise in state-space model is assumed to be Gaussian. The details of EKF can be found in [1, 14, 23, 31, 34].

However, a problem of EKF is that the nonlinear state-space equation is linearized based on the estimated values $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_i^-$, therefore the linearization leads to the propagation of Gaussian noise, which makes the performance degrade. To deal with this problem, unscented Kalman filter (UKF) is proposed.

2.3.2 Unscented Kalman Filter

Unscented Kalman filter (UKF) [18] is proposed by S. J. Julier and J. K. Uhlmann in 1997. The algorithm is applying unscented transformation to the Kalman filter. Unscented transformation is an algorithm which calculate the statistics of random variables generated from nonlinear transformation. Consider a random variable $\mathbf{x}$ with $n_{\mathbf{x}}$ dimensions which is propagated through a nonlinear function $\mathbf{y} = \mathbf{f}(\mathbf{x})$. Assume that the prior density $\Pi(\mathbf{x})$ is symmetric, the mean is $(\bar{\mathbf{x}})$ and the covariance is $P_{\mathbf{x}}$, Gaussian is a special case of the assumption. The statics of $\mathbf{y}$ is calculated by forming a set called sigma vectors and weights with $(2L+1)$ sample points $\{\mathcal{X}_n, \omega_n^{(m)}, \omega_n^{(c)}\}_{n=0}^{2L}$ according to:

$$
\begin{aligned}
\mathcal{X}_0 &= \bar{\mathbf{x}} \\
\mathcal{X}_n &= \bar{\mathbf{x}} + \left(\sqrt{(L+\lambda)P_{\mathbf{x}}}\right)_n, n = 1, ..., L \\
\mathcal{X}_n &= \bar{\mathbf{x}} - \left(\sqrt{(L+\lambda)P_{\mathbf{x}}}\right)_n, n = L+1, ..., 2L
\end{aligned} \tag{2.46}
$$

$$
\begin{aligned}
\omega_0^{(m)} &= \frac{\lambda}{L+\lambda} \\
\omega_0^{(c)} &= \frac{\lambda}{L+\lambda} + 1 - \alpha^2 + \beta \\
\omega_n^{(m)} &= \omega_n^{(c)} = \frac{1}{2(L+\lambda)}, n = L+1, ..., 2L
\end{aligned} \tag{2.47}
$$

where L is set to $n_{\mathbf{x}}$ and the scaling parameter is $\lambda = \alpha^2(L+k) - L$. The spread of the sigma points around $\bar{\mathbf{x}}$ is determined by the constant α, and the constant α is usually a small and positive value ($0 \leq \alpha \leq 10^{-4}$). The second scaling parameter k is usually set to $3 - L$. The prior knowledge of the distribution of $\mathbf{x}$ is incorporated by β, which is set to 2 in Gaussian distributions. The ith column of matrix square root is $\left(\sqrt{(L+\lambda)P_{\mathbf{x}}}\right)_n$.
The sigma points and the sigma weights have the following conditions:

$$\begin{aligned} \bar{\mathbf{x}} &= \sum_{n=0}^{2L} \omega_n^{(m)} \mathcal{X}_n \\ P_{\mathbf{x}} &= \sum_{n=0}^{2L} \omega_n^{(c)} (\mathcal{X}_n - \bar{\mathbf{x}})^T \end{aligned} \tag{2.48}$$

The sigma vectors are propagated through the nonlinear function

$$\mathcal{Y}_n = \mathbf{f}(\mathcal{X}_n), n = 0, ..., 2L \tag{2.49}$$

The mean of $\mathbf{y}$ is approximated as

$$\bar{\mathbf{y}} \approx \sum_{n=0}^{2L} \omega_n^{(m)} \mathcal{Y}_n \tag{2.50}$$

The covariance of $\mathbf{y}$ can be approximated as

$$P_{\mathbf{y}} \approx \sum_{n=0}^{2L} \omega_n^{(c)} (\mathcal{Y}_n - \bar{\mathbf{y}})(\mathcal{Y}_n - \bar{\mathbf{y}})^T \tag{2.51}$$

Using the sigma vectors above, the cross covariance can be estimated and used to implement the Kalman filter. The further details of UKF are introduced in [16–18, 32, 33].

2.3.3 Other Generative Approaches

Recently, there have been some other generative approaches that produce probability models over all variables in systems and then manipulate the models to compute the

classification and regression functions.

Dynamical System Model with a Conditional Embedding (DSMCE) operator [29] and kernel Bayes' rule (KBR) algorithms [7] are two recent generative approaches. Both of them are based on the concept of conditional embedding in RKHS, and both of them treat the true output $\{\mathbf{x}_i\}$ or the mappings of them in the RKHS feature space as the hidden states. The algorithms estimate the hidden states dynamics using the assumed state-space system model or the given training set consisting of hidden states. For most of the existing generative approaches for nonlinear systems, including DSMCE and KBR, the accurate state-space system model or a clean training set consists of hidden states that are necessary. However, in most nonlinear filtering problems, both accurate state-space models and clean training sets are unavailable. To solve this problem, Zhu et al. proposed an algorithm, which will be introduced in detail in Chapter 4. The time complexity of Zhu's algorithm is $O(n^3)$, which makes it hard to do the filtering when the size of data is large. Besides, the Kalman filter can be improved using double sided kernel introduced in Chapter 3. Therefore, we developed a novel algorithm that reducing the time complexity and improving the accuracy of the filtering problem, which will be introduced in Chapter 5.

Chapter 3

Double Sided Kernel for Linear Systems [15]

Double sided kernel was initially presented in [8]. The algorithm is developed as an improvement of the classical Kalman filter, the object is solving both parameter estimation and state estimation problem of single input single output (SISO) linear time invariant (LTI) system. Double sided kernel employs forward integration, backward integration and Cauchy formula for multiple integrals, and convert a high order differential equation into an integral form with no singularities at the boundaries of the observation window.

3.1 Parameter Estimation Using Kernel Representation of Homogeneous SISO LTI systems

The estimation problem in [8] is solving a SISO LTI system with the following structure:

$$\begin{aligned} \dot{x} &= \mathbf{A}x \\ y &= \mathbf{C}x \end{aligned} \tag{3.1}$$

where the matrix **A** is in canonical form:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix} \tag{3.2}$$

and matrix **C** is:

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & \cdots \end{bmatrix} \tag{3.3}$$

Therefore, the characteristic equation of the system is:

$$y^{(n)}(t) + a_{n-1}y^{(n-1)}(t) + \cdots + a_1 y^{(1)}(t) + a_0 y(t) = 0 \tag{3.4}$$

The parameters $a_i, i = 0, ..., n-1$ are estimated by noisy measurements $z(\tau)$.
We can treat the parameter estimation of a homogeneous system as the identification of an unchanged differential invariant $\mathcal{I}$ ($\mathcal{I} \equiv 0$).

$$\mathcal{I}(t, y(t), y^{(1)}(t), \cdots, y^{(n)}(t)) = y^{(n)}(t) + a_{n-1}y^{(n-1)}(t) + \cdots + a_0 y(t), t \leq 0 \tag{3.5}$$

There is a problem of singularity at $t = 0$. However, the problem is solved in [8] using a two-sided integration which leads to an integral representation of (3.5) in RKHS. Such representation is described by the following theorem, which has been stated and proved in [8]:

Theorem 3.1.1. There exist Hilbert-Schmidt kernels K_{DS} and K^i_{DS}, $i = 1, ..., n-1$, such that y in (3.5) is reproduced in accordance with the action of the evaluation function

$$y(t) = \int_a^b K_{DS}(t, \tau) y(\tau) d\tau, \forall t \in [a, b] \tag{3.6}$$

The derivatives of y can be computed by output integration recursively, for $i = 1, ..., n-1$ and $\forall t \in [a, b]$:

$$y^{(i)}(t) = \sum_{k=0}^{i-1} b_k(t) y^{(k)}(t) + \int_a^b K_{DS}^i(t, \tau) y(\tau) d\tau \tag{3.7}$$

where $y^{(0)} \equiv y$ and $b_k(\cdot)$ are rational functions of t. Hilbert-Schmidt kernels are square integrable functions on $L^2[a, b] \times L^2[a, b]$.

3.2 Derivation of Double Sided Kernel for Fourth Order System

John's thesis [15] developed the kernel representation of the double sided kernel for fourth order system, the characteristic equation of fourth order system can be written as:

$$y^{(4)}(t) + a_3 y^{(3)}(t) + a_2 y^{(2)}(t) + a_1 y^{(1)}(t) + a_0 t(t) = 0 \tag{3.8}$$

the time interval is $[a, b]$.

Multiply (3.8) by $(\epsilon - a)^4$ and $(b - \zeta)^4$, we have (3.9) and (3.10) respectively:

$$(\epsilon-a)^4 y^{(4)}(t)+a_3(\epsilon-a)^4 y^{(3)}(t)+a_2(\epsilon-a)^4 y^{(2)}(t)+a_1(\epsilon-a)^4 y^{(1)}(t)+a_0(\epsilon-a)^4 y(t) = 0 \tag{3.9}$$

$$(b-\zeta)^4 y^{(4)}(t)+a_3(b-\zeta)^4 y^{(3)}(t)+a_2(b-\zeta)^4 y^{(2)}(t)+a_1(b-\zeta)^4 y^{(1)}(t)+a_0(b-\zeta)^4 y(t) = 0 \tag{3.10}$$

Then, (3.9) and (3.10) will be integrated on interval $[a, a + \tau]$ and $[b - \zeta, b]$ for four times. Which means that (3.8) is integrated in forward direction on interval $[a, a + \tau]$ and in backward direction on interval $[b, b - \zeta]$.

Integrate the first term of (3.9) for the first time:

$$\int_a^{a+\tau} (\epsilon - a)^4 y^{(4)}(\epsilon) \mathrm{d}\epsilon$$

$$
\begin{aligned}
=&\tau^4 y^{(3)}(a+\tau) - \left[4(\epsilon-a)^3 y^{(2)}(\epsilon)|_a^{a+\tau} - \int_a^{a+\tau} 12(\epsilon-a)^2 y^{(2)}(\epsilon)\mathrm{d}\epsilon\right] \\
=&\tau^4 y^{(3)}(a+\tau) - 4\tau^3 y^{(2)}(a+\tau) + 12(\epsilon-a)^2 y(\epsilon)|_a^{a+\tau} - \int_a^{a+\tau} 24(\epsilon-a) y(\epsilon)\mathrm{d}\epsilon \\
=&\tau^4 y^{(3)}(a+\tau) - 4\tau^3 y^{(2)}(a+\tau) + 12\tau^2 y^{(1)}(a+\tau) \\
&- 24(\epsilon-a) y(\epsilon)|_a^{a+\tau} + \int_a^{a+\tau} 24(\epsilon-a) y(\epsilon)\mathrm{d}\epsilon
\end{aligned} \tag{3.11}
$$

$$
\begin{aligned}
\int_a^{a+\tau} (\epsilon-a)^4 y^{(4)}(\epsilon)\mathrm{d}\epsilon =&\tau^4 y^{(3)}(a+\tau) - 4\tau^3 y^{(2)}(a+\tau) + 12\tau^2 y(1)(a+\tau) \\
&- 24\tau y(a+\tau) + \int_a^{a+\tau} 24 y(\epsilon)\mathrm{d}\epsilon
\end{aligned} \tag{3.12}
$$

The upper bound of the integral is a dummy value. For convenience, denote it as $\epsilon' = a+\tau$ and therefore,

$$
\begin{aligned}
\tau^4 y^{(3)}(a+\tau) &= (\epsilon'-a)^4 y^{(3)}(\epsilon') \\
\tau^3 y^{(2)}(a+\tau) &= (\epsilon'-a)^3 y^{(2)}(\epsilon') \\
\tau^2 y^{(1)}(a+\tau) &= (\epsilon'-a)^2 y^{(1)}(\epsilon') \\
\tau y(a+\tau) &= (\epsilon'-a) y(\epsilon')
\end{aligned}
$$

Integrate the first term of (3.9) for the second time and bring in the above denotations, we have:

$$
\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon'} (\epsilon-a)^4 y^{(4)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon' \\
&\quad = \int_a^{a+\tau} (\epsilon'-a)^4 y^{(3)}(\epsilon')\mathrm{d}\epsilon' - 4\int_a^{a+\epsilon} (\epsilon'-a)^3 y^{(2)}(\epsilon')\mathrm{d}\epsilon'
\end{aligned}
$$

$$
\begin{aligned}
&+12\int_a^{a+\epsilon}(\epsilon'-a)^2y^{(1)}(\epsilon')\mathrm{d}\epsilon'-24\int_a^{a+\epsilon}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon\\
&+24\int_a^{a+\epsilon}\int_a^{\epsilon'}y(\epsilon')\mathrm{d}\epsilon\mathrm{d}\epsilon'\\
=&(\epsilon'-a)^4y^{(2)}(\epsilon')|_a^{a+\tau}-4\int_a^{a+\tau}(\epsilon'-a)^3y^{(2)}(\epsilon')\mathrm{d}\epsilon'-4(\epsilon'-a)^3y(\epsilon')|_a^{a+\tau}\\
&+12\int_a^{a+\tau}(\epsilon'-a)^2y(\epsilon')\mathrm{d}\epsilon'+12(\epsilon'-a)^2y(\epsilon')|_a^{a+\tau}\\
&-24\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'+24\int_a^{a+\epsilon}\int_a^{\epsilon'}y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\\
&-24\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'\\
=&\tau^4y^{(2)}(a+\tau)-4(\epsilon'-a)^3y(\epsilon')|_a^{a+\tau}+12\int_a^{a+\tau}(\epsilon'-a)^2y(\epsilon')\mathrm{d}\epsilon'\\
&-4\tau^3y(a+\tau)+12(\epsilon'-a)^2y(\epsilon')|_a^{a+\tau}-24\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'\\
&+12\tau^2y(a+\tau)+24\int_a^{a+\epsilon}\int_a^{\epsilon'}y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'-48\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'\\
=&\tau^4y^{(2)}(a+\tau)-4\tau^3y(a+\tau)+12(\epsilon'-a)^2y(\epsilon)|_a^{a+\tau}\\
&-24\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'-4\tau^3y(a+\tau)+24\tau^2y(a+\tau)\\
&+24\int_a^{a+\epsilon}\int_a^{\epsilon'}y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'-72\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'
\end{aligned}
\tag{3.13}
$$

And finally, we have

$$\begin{aligned}\int_a^{a+\tau}\int_a^{\epsilon'}(\epsilon-a)^4y^{(4)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon' =&\tau^4y^{(2)}(a+\tau)-8\tau^3y(a+\tau)+36\tau^2y(a+\tau)\\ &-96\int_a^{a+\tau}(\epsilon'-a)y(\epsilon')\mathrm{d}\epsilon'+24\int_a^{a+\tau}\int_a^{\epsilon'}y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\end{aligned} \tag{3.14}$$

Integrate the first term for the third time. Again, we set a dummy value $\epsilon''=a+\tau$, we have:

$$\begin{aligned}&\int_a^{a+\tau}\int_a^{\epsilon'}\int_a^{\epsilon''}(\epsilon-a)^4y^{(4)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\\ &\quad=\int_a^{a+\tau}(\epsilon''-a)^4y^{(2)}(\epsilon'')\mathrm{d}\epsilon''-8\int_a^{a+\tau}(\epsilon''-a)^3y'(\epsilon'')\mathrm{d}\epsilon''\\ &\qquad+36\int_a^{a+\tau}(\epsilon''-a)^2y(\epsilon'')\mathrm{d}\epsilon''-96\int_a^{a+\tau}\int_a^{\epsilon''}(\epsilon'-a)y(\epsilon)\mathrm{d}\epsilon'\mathrm{d}\epsilon''\\ &\qquad+24\int_a^{a+\tau}\int_a^{\epsilon'}\int_a^{\epsilon''}y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\\ &\quad=\tau^4y^{(1)}(a+\tau)-4(\epsilon''-a)^3y(\epsilon'')|_a^{a+\tau}+12\int_a^{a+\tau}(\epsilon''-a)^2y(\epsilon'')\mathrm{d}\epsilon''\\ &\qquad-8\tau^3y(a+\tau)+60\int_a^{a+\tau}(\epsilon''-a)^2y(\epsilon'')\mathrm{d}\epsilon''\\ &\qquad-96\int_a^{a+\tau}\int_a^{\epsilon''}(\epsilon'-a)y(\epsilon)\mathrm{d}\epsilon'\mathrm{d}\epsilon''+24\int_a^{a+\tau}\int_a^{\epsilon'}\int_a^{\epsilon''}y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\end{aligned} \tag{3.15}$$

Finally, we have

$$\int_a^{a+\tau}\int_a^{\epsilon'}\int_a^{\epsilon''}(\epsilon-a)^4y^{(4)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''$$

$$
\begin{aligned}
=&\tau^4 y(a+\tau) - 12\tau^3 y(a+\tau) + 72 \int_a^{a+\tau} (\epsilon'' - a)^2 y(\epsilon'') \mathrm{d}\epsilon'' \\
&- 96 \int_a^{a+\tau} \int_a^{\epsilon''} (\epsilon' - a) y(\epsilon') \mathrm{d}\epsilon \mathrm{d}\epsilon'' \\
&+ 24 \int_a^{a+\tau} \int_a^{\epsilon'} \int_a^{\epsilon''} y(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \mathrm{d}\epsilon''
\end{aligned} \tag{3.16}
$$

Integrate the first term for the fourth time, set a dummy value $\epsilon''' = a + \tau$, we have:

$$
\begin{aligned}
&\int_a^{a+\tau} \int_a^{\epsilon'''} \int_a^{\epsilon''} \int_a^{\epsilon'} (\epsilon - a)^4 y^{(4)}(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \mathrm{d}\epsilon'' \mathrm{d}\epsilon''' \\
&\quad = \int_a^{a+\tau} (\epsilon''' - a)^4 y^{(1)}(\epsilon''') \mathrm{d}\epsilon''' - \int_a^{a+\tau} 12(\epsilon''' - a)^3 y(\epsilon''') \mathrm{d}\epsilon''' \\
&\qquad + \int_a^{a+\tau} \int_a^{\epsilon'''} 72(\epsilon'' - a)^2 y(\epsilon'') \mathrm{d}\epsilon'' \mathrm{d}\epsilon''' \\
&\qquad - \int_a^{a+\tau} \int_a^{\epsilon'''} \int_a^{\epsilon''} 96(\epsilon' - a) y(\epsilon') \mathrm{d}\epsilon' \mathrm{d}\epsilon'' \mathrm{d}\epsilon''' \\
&\qquad + \int_a^{a+\tau} \int_a^{\epsilon'''} \int_a^{\epsilon''} \int_a^{\epsilon'} 24 y(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \mathrm{d}\epsilon'' \mathrm{d}\epsilon'''
\end{aligned} \tag{3.17}
$$

And finally,

$$
\begin{aligned}
&\int_a^{a+\tau} \int_a^{\epsilon'''} \int_a^{\epsilon''} \int_a^{\epsilon'} (\epsilon - a)^4 y^{(4)}(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \mathrm{d}\epsilon'' \mathrm{d}\epsilon''' \\
&\qquad = \tau^4 y(a+\tau) - \int_a^{a+\tau} 16(\epsilon''' - a)^3 y(\epsilon''') \mathrm{d}\epsilon''' \\
&\qquad\quad + \int_a^{a+\tau} \int_a^{\epsilon'''} 72(\epsilon'' - a)^2 y(\epsilon'') \mathrm{d}\epsilon'' \mathrm{d}\epsilon'''
\end{aligned}
$$

$$
\begin{aligned}
&- \int_a^{a+\tau} \int_a^{\epsilon'''} \int_a^{\epsilon''} 96(\epsilon' - a) y(\epsilon') \mathrm{d}\epsilon' \mathrm{d}\epsilon'' \mathrm{d}\epsilon''' \\
&+ \int_a^{a+\tau} \int_a^{\epsilon'''} \int_a^{\epsilon''} \int_a^{\epsilon'} 24 y(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \mathrm{d}\epsilon'' \mathrm{d}\epsilon'''
\end{aligned} \tag{3.18}
$$

Then, integrate the second term of (3.9) for the first time:

$$
\begin{aligned}
&\int_a^{a+\tau} a_3(\epsilon - a)^4 y^{(3)}(\epsilon) \mathrm{d}\epsilon \\
&= a_3(\epsilon - a)^4 y^{(2)}(\epsilon)|_a^{a+\tau} - \int_a^{a+\tau} 4a_3(\epsilon - a)^3 y^{(2)}(\epsilon) \mathrm{d}\epsilon \\
&= a_3 \tau^4 y^{(2)}(a+\tau) - \left[4a_3(\epsilon - a)^3 y^{(1)}(\epsilon)|_a^{a+\tau} - \int_a^{a+\tau} 12a_3(\epsilon - a)^2 y^{(1)}(\epsilon) \mathrm{d}\epsilon \right] \\
&= a_3 \tau^4 y^{(2)}(a+\tau) - 4a_3 \tau^3 y^{(1)}(a+\tau) + 12a_3 \tau^2 y(a+\tau) - \int_a^{a+\tau} 24a_3(\epsilon - a) y(\epsilon) \mathrm{d}\epsilon
\end{aligned} \tag{3.19}
$$

Similar to previous steps, introducing a dummy value $\epsilon' = a + \tau$ and integrate the second term for the second time:

$$
\begin{aligned}
&\int_a^{a+\tau} \int_a^{\epsilon'} a_3(\epsilon - a)^4 y^{(3)}(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \\
&\quad = \int_a^{a+\tau} a_3(\epsilon' - a)^4 y^{(2)}(\epsilon') \mathrm{d}\epsilon' - \int_a^{a+\tau} 4a_3(\epsilon' - a)^3 y^{(1)}(\epsilon') \mathrm{d}\epsilon' \\
&\qquad + \int_a^{a+\tau} 12a_3(\epsilon' - a)^2 y(\epsilon') \mathrm{d}\epsilon' - \int_a^{a+\tau} \int_a^{\epsilon'} 24a_3(\epsilon - a) y(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon' \\
&\quad = a_3 \tau^4 y^{(1)}(a+\tau) - 8a_3 \tau^3 y(a+\tau) + \int_a^{a+\tau} 36a_3(\epsilon' - a)^2 y(\epsilon') \mathrm{d}\epsilon' \\
&\qquad - \int_a^{a+\tau} \int_a^{\epsilon'} 24a_3(\epsilon - a) y(\epsilon) \mathrm{d}\epsilon \mathrm{d}\epsilon'
\end{aligned} \tag{3.20}
$$

Integrate the second term for the third time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} a_3(\epsilon-a)^4 y^{(3)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\quad= \int_a^{a+\tau} a_3(\epsilon''-a)^4 y^{(1)}(\epsilon'')\mathrm{d}\epsilon'' - \int_a^{a+\tau} 8a_3(\epsilon''-a)^3 y(\epsilon'')\mathrm{d}\epsilon'' \\
&\qquad+ \int_a^{a+\tau}\int_a^{\epsilon''} 36a_3(\epsilon'-a)^2 y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\qquad- \int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} 24a_3(\epsilon-a) y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\quad= a_3\tau^4 y(a+\tau) - \int_a^{a+\tau} 12a_3(\epsilon''-a)^3 y(\epsilon'')\mathrm{d}\epsilon'' \\
&\qquad+ \int_a^{a+\tau}\int_a^{\epsilon''} 36a_3(\epsilon'-a)^2 y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\qquad- \int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} 24a_3(\epsilon-a) y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''
\end{aligned} \tag{3.21}$$

Integrate the second term for the fourth time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} a_3(\epsilon-a)^4 y^{(3)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\quad= \int_a^{a+\tau} a_3(\epsilon'''-a)^4 y(\epsilon''')\mathrm{d}\epsilon''' - \int_a^{a-\tau}\int_a^{\epsilon'''} 12a_3(\epsilon''-a)^3 y(\epsilon'')\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\qquad+ \int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''} 36a_3(\epsilon'-a)^2 y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\qquad- \int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} 24a_3(\epsilon-a) y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon'''
\end{aligned} \tag{3.22}$$

Integrate the third term of (3.9) for the first time:

$$\begin{aligned}
&\int_a^{a+\tau} a_2(\epsilon-a)^4 y^{(2)}(\epsilon)\mathrm{d}\epsilon \\
=&a_2(\epsilon-a)^4 y^{(1)}(\epsilon)|_a^{a+\tau} - \int_a^{a+\tau} 4a_2(\epsilon-a)^3 y^{(1)}(\epsilon)\mathrm{d}\epsilon \\
=&a_2\tau^4 y^{(1)}(a+\tau) - \left[4a_2(\epsilon-a)^3 y(\epsilon)|_a^{a+\tau} - \int_a^{a+\tau} 12a_2(\epsilon-a)^2 y(\epsilon)\mathrm{d}\epsilon\right] \\
=&a_2\tau^4 y^{(1)}(a+\tau) - 4a_2\tau^3 y(a+\tau) + \int_a^{a+\tau} 12a_2(\epsilon-a)^2 y(\epsilon)\mathrm{d}\epsilon
\end{aligned} \tag{3.23}$$

Integrate the third time for the second time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon'} a_2(\epsilon-a)^4 y^{(2)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon' \\
&\quad= \int_a^{a+\tau} a_2(\epsilon'-a)^4 y^{(1)}(\epsilon')\mathrm{d}\epsilon' - \int_a^{a+\tau} 4a_2(\epsilon'-a)^3 y(\epsilon')\mathrm{d}\epsilon' \\
&\qquad+ \int_a^{a+\tau}\int_a^{\epsilon'} 12a_2(\epsilon-a)^2 y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon' \\
&=a_2\tau^4 y(a+\tau) - \int_a^{a+\tau} 8a_2(\epsilon'-a)^3 y(\epsilon')\mathrm{d}\epsilon' + \int_a^{a+\tau}\int_a^{\epsilon'} 12a_2(\epsilon-a)^2 y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'
\end{aligned} \tag{3.24}$$

Integrating the third term for the third time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} a_2(\epsilon-a)^4 y^{(2)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\qquad= \int_a^{a+\tau} a_2(\epsilon''-a)^4 y(\epsilon'')\mathrm{d}\epsilon'' - \int_a^{a+\tau}\int_a^{\epsilon''} 8a_2(\epsilon'-a)^3 y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon''
\end{aligned}$$

$$+\int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} 12a_2(\epsilon-a)^2y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \tag{3.25}$$

Integrate the third term for the fourth time:

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} a_2(\epsilon-a)^4y^{(2)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\quad=\int_a^{a+\tau}\int_a^{\epsilon'''} a_2(\epsilon''-a)^4y(\epsilon'')\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\quad\quad-\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''} 8a_2(\epsilon'-a)^3y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\quad\quad+\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} 12a_2(\epsilon-a)^2y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon'''
\end{aligned} \tag{3.26}$$

Integrate the fourth term of (3.9) for the first time,

$$\begin{aligned}
&\int_a^{a+\tau} a_1(\epsilon-a)y^{(1)}(\epsilon)\mathrm{d}\epsilon \\
&=a_1(\epsilon-a)^4y(\epsilon)|_a^{a+\tau}-\int_a^{a+\tau} 4a_1(\epsilon-a)^3y(\epsilon)\mathrm{d}\epsilon \\
&=a_1\tau^4y(a+\tau)-\int_a^{a+\tau} 4a_1(\epsilon-a)^3y(\epsilon)\mathrm{d}\epsilon
\end{aligned} \tag{3.27}$$

Integrate the fourth term for the second time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon'} a_1(\epsilon-a)^4y^{(1)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon' \\
&\quad=\int_a^{a+\tau} a_1(\epsilon'-a)^4y(\epsilon')\mathrm{d}\epsilon'-\int_a^{a+\tau}\int_a^{\epsilon'} 4a_1(\epsilon-a)^3y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'
\end{aligned} \tag{3.28}$$

Integrate the fourth term for the third time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} a_1(\epsilon-a)^4 y^{(1)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\qquad = \int_a^{a+\tau}\int_a^{\epsilon''} a_1(\epsilon'-a)^4 y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon'' \\
&\qquad\quad - \int_a^{a+\tau}\int_a^{\epsilon''}\int_a^{\epsilon'} 4a_1(\epsilon-a)^3 y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''
\end{aligned} \tag{3.29}$$

Integrate the fourth term for the fourth time,

$$\begin{aligned}
&\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} a_1(\epsilon-a)^4 y^{(1)}(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\qquad = \int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''} a_1(\epsilon'-a)^4 y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&\qquad\quad - \int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} 4a_1(\epsilon-a)^3 y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon'''
\end{aligned} \tag{3.30}$$

Integrate the last term of (3.9) for four times, we have:

$$\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'} a_0(\epsilon-a)^4 y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \tag{3.31}$$

Combining all the terms being integrated four times, we have:

$$\begin{aligned}
\tau^4 y(a+\tau) = &\int_a^{a+\tau} \left[16(\epsilon'''-a)^3 - a_3(\epsilon'''-a)^4\right] y(\epsilon''')\mathrm{d}\epsilon''' \\
&+ \int_a^{a+\tau}\int_a^{\epsilon'''} \left[-72(\epsilon''-a)^2 + 12a_3(\epsilon''-a)^3 - a_2(\epsilon''-a)^4\right] y(\epsilon'')\mathrm{d}\epsilon''\mathrm{d}\epsilon'''
\end{aligned}$$

$$
\begin{aligned}
&+\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\Big[96(\epsilon'-a)-36a_3(\epsilon'-a)^2 \\
&\qquad\qquad +8a_2(\epsilon'-a)^3-a_1(\epsilon'-a)^4\Big]y(\epsilon')\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \\
&+\int_a^{a+\tau}\int_a^{\epsilon'''}\int_a^{\epsilon''}\int_a^{\epsilon'}\Big[-24+24a_3(\epsilon-a)-12a_2(\epsilon-a)^2 \\
&\qquad\qquad +4a_1(\epsilon-a)^3-a_0(\epsilon-a)^4\Big]y(\epsilon)\mathrm{d}\epsilon\mathrm{d}\epsilon'\mathrm{d}\epsilon''\mathrm{d}\epsilon''' \qquad (3.32)
\end{aligned}
$$

In order to simplify the above equation, Cauchy's integration formula is used:

Theorem 3.2.1. Cauchy's integration formula

Suppose f is a continuous function, the $n\mathrm{th}$ repeated integral of f start from a is:

$$
f^{(-n)}(x)=\int_a^x\int_a^{\zeta_1}\cdots\int_a^{\zeta_{n-1}}f(\zeta_n)\mathrm{d}\zeta_n\cdots\mathrm{d}\zeta_2\mathrm{d}\zeta_1 \qquad (3.33)
$$

which is equivalent to

$$
f^{(-n)}(x)=\frac{1}{(n-1)!}\int_a^x(x-t)^{n-1}f(t)\mathrm{d}t \qquad (3.34)
$$

In (3.32), let $a+\tau=t$ and apply Cauchy's integration formula,

$$
(t-a)^4y(t)\triangleq\int_a^t K_{F,y}(t,\tau)y(\tau)\,\mathrm{d}\tau \qquad (3.35)
$$

Where $K_{F,y}(t,\tau)$ is

$$
\begin{aligned}
K_{F,y}(t,\tau)=&\Big[16(\tau-a)^3-a_3(\tau-a)^4\Big] \\
&+(t-\tau)\Big[-72(\tau-a)^2+12a_3(\tau-a)^3-a_2(\tau-a)^4\Big] \\
&+\frac{(t-\tau)^2}{2}\Big[96(\tau-a)-36a_3(\tau-a)^2+8a_2(\tau-a)^3-a_1(\tau-a)^4\Big]
\end{aligned}
$$

$$+\frac{(t-\tau)^3}{6}\Big[-24+24a_3(\tau-a)-12a_2(\tau-a)^2+4a_1(\tau-a)^3-a_0(\tau-a)^4\Big] \tag{3.36}$$

Now consider (3.10), integrate the first term of it for the first time:

$$\begin{aligned}
&\int_{b-\sigma}^{b}(b-\zeta)^4y^{(4)}(\zeta)\mathrm{d}\zeta\\
=&(b-\zeta)^4y^{(3)}\zeta\mid_{b-\sigma}^{b}+\int_{b-\sigma}^{b}4(b-\zeta)^3y^{(3)}(\zeta)\mathrm{d}\zeta\\
=&-\zeta^4y^{(3)}(b-\sigma)+\Big[4(b-\zeta)^3y^{(2)}(\zeta)\mid_{b-\sigma}^{b}+\int_{b-\sigma}^{b}12(b-\zeta)^2y^{(2)}(\zeta)\mathrm{d}\zeta\Big]\\
=&-\zeta^4y^{(3)}(b-\sigma)-4\zeta^3y^{(2)}(b-\sigma)+\Big[12(b-\zeta)^2y^{(1)}(\zeta)\mid_{b-\sigma}^{b}+\int_{b-\sigma}^{b}24(b-\zeta)y^{(1)}(\zeta)\mathrm{d}\zeta\Big]\\
=&-\zeta^4y^{(3)}(b-\sigma)-4\zeta^3y^{(2)}(b-\sigma)-12\zeta^2y^{(1)}(b-\sigma)\\
&+\Big[24(b-\zeta)y(\zeta)\mid_{b-\sigma}^{b}+\int_{b-\sigma}^{b}24y(\zeta)\mathrm{d}\zeta\Big]\\
=&-\zeta^4y^{(3)}(b-\sigma)-4\zeta^3y^{(2)}(b-\sigma)-12\zeta^2y^{(1)}(b-\sigma)-24\zeta y(b-\sigma)+\int_{b-\sigma}^{b}24y(\zeta)\mathrm{d}\zeta
\end{aligned} \tag{3.37}$$

Similar to forward integration, the upper bound of the integral in backward integration is also replaced by a dummy value $\zeta'=b-\zeta$, we have

$$\begin{aligned}
-\sigma^4y^{(3)}(b-\sigma)&=-(b-\zeta')^4y^{(3)}(\zeta')\\
-4\sigma^3y^{(2)}(b-\sigma)&=-4(b-\zeta')^3y^{(2)}(\zeta')\\
-12\sigma^2y^{(1)}(b-\sigma)&=-12(b-\zeta')^2y^{(1)}(\zeta')\\
-24\sigma y(b-\sigma)&=-24(b-\zeta')y(\zeta')
\end{aligned}$$

Integrate the first term of (3.10) for the second time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta'}^{b}(b-\zeta)^4y^{(4)}(\zeta)\mathrm{d}\zeta \\
&\quad=-\int_{b-\sigma}^{b}(b-\zeta)^4y^{(3)}(\zeta')\mathrm{d}\zeta'-\int_{b-\sigma}^{b}4(b-\zeta')^3y^{(2)}(\zeta')\mathrm{d}\zeta' \\
&\qquad-\int_{b-\sigma}^{b}12(b-\zeta')^2y^{(1)}(\zeta')\mathrm{d}\zeta'-\int_{b-\sigma}^{b}24(b-\zeta')y(\zeta')\mathrm{d}\zeta'+\int_{b-\sigma}^{b}\int_{\zeta'}^{b}24y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta' \\
&\quad=\zeta^4y^{(2)}(b-\sigma)+8\zeta^3y^{(1)}(b-\sigma)+36\zeta^2y(b-\sigma) \\
&\qquad-\int_{b-\sigma}^{b}96(b-\zeta')y(\zeta')\mathrm{d}\zeta'+\int_{b-\sigma}^{b}\int_{\zeta'}^{b}24y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'
\end{aligned}\tag{3.38}$$

Set dummy value to $\zeta''=b-\sigma$, integrate the first term for the third time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}(b-\zeta)^4y^{(4)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\quad=\int_{b-\sigma}^{b}(b-\zeta'')^4y^{(2)}(\zeta'')\mathrm{d}\zeta''+\int_{b-\sigma}^{b}8(b-\zeta'')^3y^{(1)}(\zeta'')\mathrm{d}\zeta''+\int_{b-\sigma}^{b}36(b-\zeta'')^2y(\zeta'')\mathrm{d}\zeta'' \\
&\qquad-\int_{b-\sigma}^{b}\int_{\zeta''}^{b}96(b-\zeta')y(\zeta')\mathrm{d}\zeta'+\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}24y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\quad=-\zeta^4y^{(1)}(b-\sigma)-12\zeta^3y(b-\sigma)+\int_{b-\sigma}^{b}72(b-\zeta'')^2y(\zeta'')\mathrm{d}\zeta'' \\
&\qquad-\int_{b-\sigma}^{b}\int_{\zeta''}^{b}96(b-\zeta')y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''+\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}24y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''
\end{aligned}\tag{3.39}$$

Set the dummy value to $\zeta'''=b-\sigma$, then integrate the first term for the fourth time,

$$\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}(b-\zeta)^4y^{(4)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''$$

$$
\begin{aligned}
&= -\int_{b-\sigma}^{b}(b-\zeta''')^4 y^{(1)}(\zeta''')\mathrm{d}\zeta''' - \int_{b-\sigma}^{b} 12(b-\zeta''')^3 y(\zeta''')\mathrm{d}\zeta''' \\
&\quad + \int_{b-\sigma}^{b}\int_{\zeta'''}^{b} 72(b-\zeta'')^2 y(\zeta'')\mathrm{d}\zeta''\zeta''' - \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b} 96(b-\zeta')y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \\
&\quad + \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} 24y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \qquad (3.40)
\end{aligned}
$$

Integrate the second term of (3.10) for the first time,

$$
\begin{aligned}
&\int_{b-\sigma}^{b} a_3(b-\zeta)^4 y^{(3)}(\zeta)\mathrm{d}\zeta \\
&= a_3(b-\zeta)^4 y^{(2)}(\zeta)\ |_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 4a_3(b-\zeta)^2 y^{(2)}(\zeta)\mathrm{d}\zeta \\
&= -a_3\sigma^4 y^{(2)}(b-\sigma) + \left[4a_3(b-\zeta)y^{(1)}(\zeta)\ |_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 12a_3(b-\zeta)^2 y^{(1)}(\zeta)\mathrm{d}\zeta\right] \\
&= -a_3\sigma^4 y^{(2)}(b-\sigma) - 4a_3\sigma^3 y^{(1)}(b-\sigma) - 12a_3\sigma^2 y(b-\sigma) \\
&\quad + \int_{b-\sigma}^{b} 24a_3(b-\zeta)y(\zeta)\mathrm{d}\zeta \qquad (3.41)
\end{aligned}
$$

Set the dummy value $\zeta' = b - \sigma$, integrating the second term for the second time,

$$
\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta'}^{b} a_3(b-\zeta)^4 y^3(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta' \\
&\quad = -\int_{b-\sigma}^{b} a_3(b-\zeta')^4 y^{(2)}(\zeta')\mathrm{d}\zeta' - \int_{b-\sigma}^{b} 4a_3(b-\zeta')^3 y^{(1)}(\zeta')\mathrm{d}\zeta' \\
&\qquad - \int_{b-\sigma}^{b} 12a_3(b-\zeta')^2 y(\zeta')\mathrm{d}\zeta' + \int_{b-\sigma}^{b}\int_{\zeta'}^{b} 24a_3(b-\zeta)y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'
\end{aligned}
$$

$$=a_3\sigma^4 y^{(1)}(b-\sigma)+8a_3\sigma^{(3)}y(b-\sigma)-\int_{b-\sigma}^{b}36a_3(b-\zeta')^2y(\zeta')\mathrm{d}\zeta'$$
$$+\int_{b-\sigma}^{b}\int_{\zeta'}^{b}24a_3(b-\zeta)y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta' \tag{3.42}$$

Integrating the second term for the third time,

$$\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}a_3(b-\zeta)^4y^3(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''$$
$$=\int_{b-\sigma}^{b}a_3(b-\zeta'')^4y^{(1)}(\zeta'')\mathrm{d}\zeta''+\int_{b-\sigma}^{b}8a_3(b-\zeta'')^3y(\zeta'')\mathrm{d}\zeta''$$
$$-\int_{b-\sigma}^{b}\int_{\zeta''}^{b}36a_3(b-\zeta')^2y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''+\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}24a_3(b-\zeta)y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''$$
$$=-a_3\sigma^4y(b-\sigma)+\int_{b-\sigma}^{b}12a_3(b-\zeta'')^3y(\zeta'')\mathrm{d}\zeta''-\int_{b-\sigma}^{b}\int_{\zeta''}^{b}36a_3(b-\zeta')^2y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''$$
$$+\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}24a_3(b-\zeta)y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \tag{3.43}$$

Integrate the second term for the fourth time,

$$\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}a_3(b-\zeta)^4y^3(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''$$
$$=-\int_{b-\sigma}^{b}a_3(b-\zeta''')^4y(\zeta''')\mathrm{d}\zeta'''+\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}12a_3(b-\zeta'')^3y(\zeta'')\mathrm{d}\zeta''\mathrm{d}\zeta'''$$
$$-\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}36a_3(b-\zeta')^2y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''$$
$$+\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}24a_3(b-\zeta)y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta''\mathrm{d}\zeta''\mathrm{d}\zeta''' \tag{3.44}$$

Integrate the third term of (3.10) for the first time, we have

$$\begin{aligned}
&\int_{b-\sigma}^{b} a_2(b-\zeta)^4 y^{(2)}(\zeta)\mathrm{d}\zeta \\
=&a_2(b-\zeta)^4 y^{(1)}(\zeta)\mid_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 4a_2(b-\zeta)^3 y^{(1)}(\zeta)\mathrm{d}\zeta \\
=&-a_2\sigma^4 y^{(1)}(b-\sigma) + \left[4a_2(b-\zeta)^3 y(\zeta)\mid_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 12a_2(b-\zeta)^2 y(\zeta)\mathrm{d}\zeta\right] \\
=&-a_2\sigma^4 y^{(1)}(b-\sigma) - 4a_2\sigma^3 y(b-\sigma) + \int_{b-\sigma}^{b} 12a_2(b-\zeta)^2 y(\zeta)\mathrm{d}\zeta
\end{aligned} \tag{3.45}$$

Integrate the third term for the second time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta'}^{b} a_2(b-\zeta')^4 y^{(2)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta' \\
=&-\int_{b-\sigma}^{b} a_2(b-\zeta')^4 y^{(1)}(\zeta') - \int_{b-\sigma}^{b} 4a_2(b-\zeta')^3 y(\zeta')\mathrm{d}\zeta' + \int_{b-\sigma}^{b}\int_{\zeta'}^{b} 12a_2(b-\zeta')^2 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta' \\
=&\, a_2\sigma^4 y(b-\sigma) - \int_{b-\sigma}^{b} 8a_2(b-\zeta')^3 y(\zeta')\mathrm{d}\zeta' + \int_{b-\sigma}^{b}\int_{\zeta'}^{b} 12a_2(b-\zeta)^2 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'
\end{aligned} \tag{3.46}$$

Integrate the third term for the third time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} a_2(b-\zeta)^4 y^{(2)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\quad=\int_{b-\sigma}^{b} a_2(b-\zeta'')^{(4)} y(\zeta'')\mathrm{d}\zeta'' - \int_{b-\sigma}^{b}\int_{\zeta''}^{b} 8a_2(b-\zeta')^3 y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\qquad+\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} 12a_2(b-\zeta)^2 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''
\end{aligned} \tag{3.47}$$

Integrate the third term for the fourth time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} a_2(b-\zeta)^4 y^{(2)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \\
&\qquad = \int_{b-\sigma}^{b}\int_{\zeta'''}^{b} a_2(b-\zeta'')^4 y(\zeta'')\mathrm{d}\zeta''\mathrm{d}\zeta''' - \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b} 8a_2(b-\zeta')^3 y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \\
&\qquad\quad + \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} 12a_2(b-\zeta)^2 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''
\end{aligned} \tag{3.48}$$

Integrate the fourth term for the first time,

$$\begin{aligned}
\int_{b-\sigma}^{b} a_1(b-\zeta)^4 y^{(1)}(\zeta)\mathrm{d}\zeta &= a_1(b-\zeta)^4 y(\zeta)\mid_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 4a_1(b-\zeta)^3 y(\zeta)\mathrm{d}\zeta \\
&= -a_1\sigma^4 y(b-\sigma) + \int_{b-\sigma}^{b} 4a_1(b-\zeta)^3 y(\zeta)\mathrm{d}\zeta
\end{aligned} \tag{3.49}$$

Integrate the fourth term for the second time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta'}^{b} a_1(b-\zeta)^4 y^{(1)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta' \\
&\qquad = -\int_{b-\sigma}^{b} a_1(b-\zeta')^4 y(\zeta')\mathrm{d}\zeta' + \int_{b-\sigma}^{b}\int_{\zeta'}^{b} 4a_1(b-\zeta)^3 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'
\end{aligned} \tag{3.50}$$

Integrate the fourth term for the third time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} a_1(b-\zeta)^4 y^{(1)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\qquad = -\int_{b-\sigma}^{b}\int_{\zeta''}^{b} a_1(b-\zeta')^4 y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta'' + \int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} 4a_1(b-\zeta)^3 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''
\end{aligned} \tag{3.51}$$

Integrate the fourth term for the last time,

$$\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} a_1(b-\zeta)^4 y^{(1)}(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''$$
$$= -\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b} a_1(b-\zeta')^4 y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' + \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} 4a_1(b-\zeta)^3 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \tag{3.52}$$

Integrate the last term of (3.10) four times, we have

$$\int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b} a_0(b-\zeta)^4 y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \tag{3.53}$$

Combining all the integration of the five terms from (3.10) above, we have

$$\begin{aligned}
&\sigma^4 y(b-\sigma)\\
&= \int_{b-\sigma}^{b}\Big[16(b-\zeta''')^3 + a_3(b-\zeta''')^4\Big]y(\zeta''')\mathrm{d}\zeta'''\\
&\quad + \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\Big[-72(b-\zeta'')^2 - 12a_3(b-\zeta'')^3 - a_2(b-\zeta'')^4\Big]y(\zeta'')\zeta''\zeta'''\\
&\quad + \int_{b-\sigma}^{b}\int_{\zeta''}^{b}\int_{\zeta''}^{b}\Big[96(b-\zeta') + 36a_3(b-\zeta')^2 + 8a_2(b-\zeta')^3 + a_1(b-\zeta')^4\Big]y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''\\
&\quad + \int_{b-\sigma}^{b}\int_{\zeta'''}^{b}\int_{\zeta''}^{b}\int_{\zeta'}^{b}\Big[-24 - 24a_3(b-\zeta) - 12a_2(b-\zeta)^2\\
&\qquad\qquad - 4a_1(b-\zeta)^3 - a_0(b-\zeta)^4\Big]y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''
\end{aligned} \tag{3.54}$$

In order to apply Cauchy's integration formula on (3.54), the integration intervals are reversed, and the signs are changed, we have

$$
\begin{aligned}
&\sigma^4 y(b-\sigma) \\
&\quad = \int_b^{b-\sigma} \Big[-16(b-\zeta''')^3 - a_3(b-\zeta''')^4 \Big] y(\zeta''')\mathrm{d}\zeta''' \\
&\qquad + \int_b^{b-\sigma} \int_b^{\zeta'''} \Big[-72(b-\zeta'')^2 - 12a_3(b-\zeta'')^3 - a_2(b-\zeta'')^4 \Big] y(\zeta'')\zeta''\zeta''' \\
&\qquad + \int_b^{b-\sigma} \int_b^{\zeta''} \int_b^{\zeta''} \Big[-96(b-\zeta') - 36a_3(b-\zeta')^2 - 8a_2(b-\zeta')^3 - a_1(b-\zeta')^4 \Big] y(\zeta')\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta''' \\
&\qquad + \int_b^{b-\sigma} \int_b^{\zeta'''} \int_b^{\zeta''} \int_b^{\zeta'} \Big[-24 - 24a_3(b-\zeta) - 12a_2(b-\zeta)^2 \\
&\qquad\qquad - 4a_1(b-\zeta)^3 - a_0(b-\zeta)^4 \Big] y(\zeta)\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''\mathrm{d}\zeta'''
\end{aligned} \tag{3.55}
$$

Substitute σ into $b-t$, then apply Cauchy's integration formula (3.34) on (3.55), we have

$$
\begin{aligned}
&(b-t)^4 y(t) \\
&\quad = \int_b^t \Big[-16(b-\sigma)^3 - a_3(b-\sigma)^4 \Big] y(\sigma)\mathrm{d}\sigma \\
&\qquad + \int_b^t (t-\sigma) \Big[-72(b-\sigma)^2 - 12a_3(b-\sigma)^3 - a_2(b-\sigma)^4 \Big] y(\sigma)\mathrm{d}\sigma \\
&\qquad + \int_b^t \frac{(t-\sigma)^2}{2} \Big[-96(b-\sigma) - 36a_3(b-\sigma)^2 - 8a_2(b-\sigma)^3 - a_1(b-\sigma)^4 \Big] y(\sigma)\mathrm{d}\sigma \\
&\qquad + \int_b^t \frac{(t-\sigma)^3}{6} \Big[-24 - 24a_3(b-\sigma) - 12a_2(b-\sigma)^2 \\
&\qquad\qquad - 4a_1(b-\sigma)^3 - a_0(b-\sigma)^4 \Big] y(\sigma)\mathrm{d}\sigma
\end{aligned} \tag{3.56}
$$

Reverse the integration intervals again, and change σ into τ for convenience, we have

$$\begin{aligned}
&(b-t)^4 y(t) \\
&\quad = \int_t^b \left[16(b-\tau)^3 + a_3(b-\tau)^4\right] y(\tau)\mathrm{d}\tau \\
&\qquad + \int_t^b (t-\tau)\left[72(b-\tau)^2 + 12a_3(b-\tau)^3 + a_2(b-\tau)^4\right] y(\tau)\mathrm{d}\tau \\
&\qquad + \int_t^b \frac{(t-\tau)^2}{2}\left[96(b-\tau) + 36a_3(b-\tau)^2 + 8a_2(b-\tau)^3 + a_1(b-\tau)^4\right] y(\tau)\mathrm{d}\tau \\
&\qquad + \int_t^b \frac{(t-\tau)^3}{6}\left[24 + 24a_3(b-\tau) + 12a_2(b-\tau)^2 + 4a_1(b-\tau)^3 + a_0(b-\tau)^4\right] y(\tau)\mathrm{d}\tau
\end{aligned} \tag{3.57}$$

Then, we have

$$(b-t)^4 y(t) \triangleq \int_t^b K_{B,y}(t,\tau) y(\tau)\,\mathrm{d}\tau \tag{3.58}$$

where

$$\begin{aligned}
K_{B,y}(t,\tau) = &\left[16(b-\tau)^3 + a_3(b-\tau)^4\right] \\
&+ (t-\tau)\left[72(b-\tau)^2 + 12a_3(b-\tau)^3 + a_2(b-\tau)^4\right] \\
&+ \frac{(t-\tau)^2}{2}\left[96(b-\tau) + 36a_3(b-\tau)^2 + 8a_2(b-\tau)^3 + a_1(b-\tau)^4\right] \\
&+ \frac{(t-\tau)^3}{6}\left[24 + 24a_3(b-\tau) + 12a_2(b-\tau)^2 + 4a_1(b-\tau)^3 + a_0(b-\tau)^4\right]
\end{aligned} \tag{3.59}$$

Then, add (3.35) and (3.58) together, and divide both sides by $[(t-a)^4 + (b-t)^4]$, we have

$$y(t) = \int_a^b K_{DS,y}(t,\tau) y(\tau)\mathrm{d}\tau \tag{3.60}$$

where

$$K_{DS,y} \triangleq \frac{1}{[(t-a)^4 + (b-t)^4]} \begin{cases} K_{F,y}(t,\tau) : \tau \leq t \\ K_{B,y}(t,\tau) : \tau > t \end{cases} \tag{3.61}$$

$K_{DS,y}$ is called the double sided kernel, which can be treated as a feature map that is mapping to a RKHS.

3.3 Parameter and State Estimation Using Double Sided Kernel

Consider the general fourth order system in (3.8),

$$y^{(4)}(t) + a_3 y^{(3)}(t) + a_2 y^{(2)}(t) + a_1 y^{(1)}(t) + a_0 y(t) = 0$$

The objective of parameter estimation is to estimate the parameter vector $a \triangleq (a_0, a_1, a_2, a_3)$ of the system above from the noisy measurements $z(\tau)$, the objective of state estimation is to estimate $y(t)$ from the noisy measurements $z(\tau)$

Suppose in noisy measurements, there are finite number n of data, and each of them corresponds to a time instance t_i, where $i = 1, ..., n$. One of the many ways to determine the parameter a is to find the optimal solution of

$$\begin{aligned} &\min\{J(a) := \frac{1}{2n}\sum_{i=1}^{n}(y(t_i) - < y, K_{DS}(t_i, \cdot) >_2)^2 \mid \text{w.r.t. } a \in \mathbb{R}^3\} \\ &= \min\{\frac{1}{2n}\sum_{i=1}^{n}\left[y(t_i) - \int_a^b K_{DS}(t_i,\tau)y(\tau)d\tau\right]^2 \mid \text{w.r.t. } a \in \mathbb{R}^3\} \end{aligned} \tag{3.62}$$

which is derived from (3.60), and the continuous version of (3.62) is

$$\min\{\frac{1}{2T}\int_a^b \left[y(t) - \int_a^b K_{DS}(t,\tau)y(\tau)d\tau\right]^2 dt \mid \text{w.r.t. } a \in \mathbb{R}^3\} \tag{3.63}$$

where $T := b - a$. Because $y(t)$ is unknown, we replace $y(t)$ with its noisy measurement $z(t)$ in cost function (3.63) during the optimization:

$$\min\{\frac{1}{2T}\int_a^b \left[z(t) - \int_a^b K_{DS}(t,\tau)z(\tau)d\tau\right]^2 dt \mid \text{w.r.t. } a \in \mathbb{R}^3\} \tag{3.64}$$

The double sided kernel can be expressed as a scalar product of parameter vector a and some partial kernels:

$$K_{DS}(t,\tau) = K_v(t,\tau)^T a + k_{v5}(t,\tau)$$

which is equivalent to

$$K_{DS}(t,\tau) = a^T K_v(t,\tau) + k_{v5}(t,\tau) \tag{3.65}$$

where

$$K_v(t,\tau)^T := [k_{v1}(t,\tau), k_{v2}(t,\tau), k_{v3}(t,\tau), k_{v5}(t,\tau)];\ a := [a_0, a_1, a_2, a_3]^T \tag{3.66}$$

Rewrite the cost function, we have

$$\begin{aligned}
J(a) :=& \frac{1}{2T}\int_a^b \left[z(t) - \int_a^b K_{DS}(t,\tau)\ z(\tau)d\tau\right]^2 dt \\
=& \frac{1}{T}\int_a^b \left[\frac{1}{2}z(t)^2 - z(t)\int_a^b K_{DS}(t,\tau)\ z(\tau)d\tau + \frac{1}{2}\left(\int_a^b K_{DS}(t,\tau)\ z(\tau)d\tau\right)^2\right] dt \\
=& \frac{1}{T}\int_a^b \left[\frac{1}{2}z(t)^2 - z(t)\int_a^b K_{DS}(t,\tau)\ z(\tau)d\tau\right. \\
&\left. + \frac{1}{2}\int_a^b K_{DS}(t,\tau)\ z(\tau)d\tau \int_a^b K_{DS}(t,s)\ z(s)ds\right] dt \\
=& \frac{1}{2T}\int_a^b z(t)^2 dt - \frac{1}{T}\int_a^b \int_a^b K_{DS}(t,\tau)\ z(\tau)z(t)\ d\tau\ dt \\
&+ \frac{1}{2T}\int_a^b \int_a^b \int_a^b K_{DS}(t,\tau)K_{DS}(t,s)\ z(\tau)z(s)\ d\tau\ ds\ dt
\end{aligned}$$

Then substitute the double sided kernel K_{DS} with (3.65), we have

$$\begin{aligned}
J(a) =&\frac{1}{2T}\int_a^b z(t)^2dt - \frac{1}{T}\int_a^b\int_a^b [K_v(t,\tau)^T a + k_{v5}(t,\tau)]\ z(\tau)z(t)\ d\tau\ dt \\
&+ \frac{1}{2T}\int_a^b\int_a^b\int_a^b [a^T K_v(t,\tau) + k_{v5}(t,\tau)][K_v(t,s)^T a + k_{v5}(t,s)]\ z(\tau)z(s)\ d\tau\ ds\ dt \\
=&\frac{1}{2T}\int_a^b z(t)^2dt - \frac{1}{T}\int_a^b\int_a^b K_v(t,\tau)^T\ z(\tau)z(t)\ d\tau\ dt\ a \\
&- \frac{1}{T}\int_a^b\int_a^b k_{v5}(t,\tau)\ z(\tau)z(t)\ d\tau\ dt \\
&+ \frac{1}{2T}\ a^T\int_a^b\int_a^b\left[\int_a^b K_v(t,\tau)K_v(t,s)^T\ dt\right] z(\tau)z(s)\ d\tau\ ds\ a \\
&+ \frac{1}{2T}\int_a^b\int_a^b\int_a^b k_{v5}(t,\tau)K_v(t,s)^T\ z(\tau)z(s)\ d\tau\ ds\ dt\ a \\
&+ a^T\frac{1}{2T}\int_a^b\int_a^b\int_a^b K_v(t,\tau)k_{v5}(t,s)\ z(\tau)z(s)\ d\tau\ ds\ dt \\
&+ \frac{1}{2T}\int_a^b\int_a^b\int_a^b k_{v5}(t,\tau)k_{v5}(t,s)]\ z(\tau)z(s)\ d\tau\ ds\ dt
\end{aligned}$$

The cost function $J(a)$ can be written as a standard quadratic

$$J(a) = d + b^T a + \frac{1}{2}a^T C a \tag{3.67}$$

where

$$\begin{aligned}
d :=& \left\{\frac{1}{2T}\int_a^b y(t)^2dt - \frac{1}{T}\int_a^b\int_a^b k_{v5}(t,\tau)\ z(\tau)z(t)\ d\tau\ dt\right. \\
&\left.+\frac{1}{2T}\int_a^b\int_a^b\int_a^b k_{v5}(t,\tau)k_{v5}(t,s)]\ z(\tau)z(s)\ d\tau\ ds\ dt\right\} \\
b^T :=& \left\{-\frac{1}{T}\int_a^b\int_a^b K_v(t,\tau)^T\ z(\tau)z(t)\ d\tau\ dt\right. \\
&\left.+\frac{1}{2T}\int_a^b\int_a^b\int_a^b [K_v(t,s)^T k_{v5}(t,\tau) + K_v(t,\tau)^T k_{v5}(t,s)]\ z(\tau)z(s)\ d\tau\ ds\ dt\right\} \\
C :=& \frac{1}{T}\left\{\int_a^b\int_a^b\left[\int_a^b K_v(t,\tau)K_v(t,s)^T\ dt\right] z(\tau)z(s)\ d\tau\ ds\right\}
\end{aligned}$$

Therefore, the minimum of the cost function $J(a)$ with respect to a is attained uniquely and globally at

$$\hat{a} = -C^{-1}b \tag{3.68}$$

And the minimum of the cost function is

$$J(\hat{a}) = d - \frac{1}{2}b^T C^{-1} b \tag{3.69}$$

Also, note that the triple integrals above can be written as alternative integral products expressions, which are easier for us to handle numerically:

$$\int_a^b \int_a^b \int_a^b k_{v5}(t,\tau)k_{v5}(t,s)\ z(\tau)z(s)\ d\tau\ ds\ dt = \int_a^b \left(\int_a^b k_{v5}(t,\tau)\ z(\tau)\ d\tau \right)^2 dt$$

$$\begin{aligned} &\int_a^b \int_a^b \int_a^b \left[K_v(t,s)^T k_{v5}(t,\tau) + K_v(t,\tau)^T k_{v5}(t,s)\right] z(\tau)z(s)\ d\tau\ ds\ dt \\ &\quad = \int_a^b \left(\int_a^b K_v(t,s)^T z(s)\ ds \right) \left(\int_a^b k_{v5}(t,\tau)\ z(\tau)\ d\tau \right)\ dt \\ &\quad\quad + \int_a^b \left(\int_a^b K_v(t,\tau)^T z(\tau)\ d\tau \right) \left(\int_a^b k_{v5}(t,s)\ z(s)\ ds \right)\ dt \end{aligned}$$

$$\begin{aligned} &\int_a^b \int_a^b \left[\int_a^b K_v(t,\tau)K_v(t,s)^T\ dt \right] z(\tau)z(s)\ d\tau\ ds \\ &\quad = \int_a^b \left[\int_a^b K_v(t,\tau)\ z(\tau)\ d\tau \right] \left[\int_a^b K_v(t,s)\ z(s)\ ds \right]^T\ dt \end{aligned}$$

The discrete form of the cost in (3.62) can also be computed in a similar way:

$$\begin{aligned} J(a) :=& \frac{1}{2n} \sum_{i=1}^{n} \left[z(t_i) - \int_a^b K_{DS}(t_i,\tau)\ z(\tau) d\tau \right]^2 \\ =& \frac{1}{n} \sum_{i=1}^{n} \left[\frac{1}{2} z(t_i)^2 - z(t_i) \int_a^b K_{DS}(t_i,\tau)\ z(\tau) d\tau + \frac{1}{2} \left(\int_a^b K_{DS}(t_i,\tau)\ z(\tau) d\tau \right)^2 \right] \end{aligned}$$

$$
\begin{aligned}
=&\frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{2}z(t_i)^2 - z(t_i)\int_a^b K_{DS}(t_i,\tau)\ z(\tau)d\tau\right.\\
&\left.+\frac{1}{2}\int_a^b K_{DS}(t_i,\tau)\ z(\tau)d\tau\int_a^b K_{DS}(t_i,s)\ z(s)ds\right]\\
=&\frac{1}{2n}\sum_{i=1}^{n}z(t_i)^2 - \frac{1}{n}\sum_{i=1}^{n}z(t_i)\int_a^b K_{DS}(t_i,\tau)\ z(\tau)\ d\tau\\
&+\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b K_{DS}(t_i,\tau)K_{DS}(t_i,s)\ z(\tau)z(s)\ d\tau\ ds
\end{aligned}
$$

Substitute double sided kernel K_{DS} with (3.65) yields

$$
\begin{aligned}
J(a) =&\frac{1}{2n}\sum_{i=1}^{n}z(t_i)^2 - \frac{1}{n}\sum_{i=1}^{n}\int_a^b\left[K_v(t_i,\tau)a + k_{v5}(t_i,\tau)\right]\ z(\tau)z(t_i)\ d\tau\\
&+\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b[a^TK_v(t_i,\tau)+k_{v5}(t_i,\tau)][K_v(t_i,s)^Ta+k_{v5}(t_i,s)]\ z(\tau)z(s)\ d\tau\ ds\\
=&\frac{1}{2n}\sum_{i=1}^{n}z(t_i)^2dt - \frac{1}{n}\sum_{i=1}^{n}\int_a^b K_v(t_i,\tau)^T\ z(\tau)z(t_i)\ d\tau\ a\\
&-\frac{1}{n}\sum_{i=1}^{n}\int_a^b k_{v5}(t_i,\tau)\ z(\tau)z(t_i)\ d\tau\\
&+\frac{1}{2n}a^T\sum_{i=1}^{n}\int_a^b\int_a^b K_v(t_i,\tau)K_v(t_i,s)^Tz(\tau)z(s)\ d\tau\ ds\ a\\
&+\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b k_{v5}(t_i,\tau)K_v(t_i,s)^T\ z(\tau)z(s)\ d\tau\ ds\ a\\
&+a^T\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b K_v(t_i,\tau)k_{v5}(t_i,s)\ z(\tau)z(s)\ d\tau\ ds\\
&+\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b k_{v5}(t_i,\tau)k_{v5}(t_i,s)]\ z(\tau)z(s)\ d\tau\ ds
\end{aligned}
$$

Rewrite cost function $J(a)$ as a standard quadratic,

$$
J(a) = d + b^Ta + \frac{1}{2}a^TCa
$$

where

$$
\begin{aligned}
d := & \left\{ \frac{1}{2n} \sum_{i=1}^{n} z(t_i)^2 - \frac{1}{n} \sum_{i=1}^{n} \int_a^b k_{v5}(t_i, \tau)\ z(\tau) z(t_i)\ d\tau \right. \\
& \left. + \frac{1}{2n} \sum_{i=1}^{n} \int_a^b \int_a^b k_{v5}(t_i, \tau) k_{v5}(t_i, s)]\ z(\tau) z(s)\ d\tau\ ds \right\} \\
b^T := & \left\{ -\frac{1}{n} \sum_{i=1}^{n} \int_a^b K_v(t_i, \tau)^T\ z(\tau) z(t_i)\ d\tau \right. \\
& \left. + \frac{1}{2n} \sum_{i=1}^{n} \int_a^b \int_a^b [K_v(t_i, s)^T k_{v5}(t_i, \tau) + K_v(t_i, \tau)^T k_{v5}(t_i, s)]\ z(\tau) z(s)\ d\tau\ ds \right\} \\
C := & \left\{ \frac{1}{n} \sum_{i=1}^{n} \int_a^b \int_a^b K_v(t_i, \tau) K_v(t_i, s)^T\ z(\tau) z(s)\ d\tau\ ds \right\}
\end{aligned}
$$

Finally, the minimum of discrete cost function $J(a)$ is attained uniquely and globally at

$$\hat{a} = -C^{-1} b \tag{3.70}$$

Then the system can be reconstructed by

$$\hat{y} = \int_a^b \hat{K}_{DS}(t_i, \tau) z(\tau) d\tau \tag{3.71}$$

where $z(\tau)$ is the noisy measurement.

Chapter 4

Kalman Filter in RKHS

In [35], Zhu et al. proposed a new algorithm that can estimate the true output signal produced by a dynamical system model such as that in (6.4). The problem is challenging because Zhu's procedure must succeed without knowing anything about the actual state-space model. Additionally, the measured system output is perturbed by additive Gaussian noise. What is implied is that the model can be nonlinear. The approach of Zhu et al. involves the use of an RKHS embedding of a noisy measurement time series $\{\mathbf{y}_i\}$ where $\mathbf{y}_i = y(t_i)$ with $\{t_i\}$ represents the corresponding sampling times of the system output. In order to explain Zhu's methodology in full detail, we first define Hilbert space embeddings in Subsection 4.1. In Subsection 4.2, we introduce a Kalman filter in the Hilbert space of embeddings which is an RKHS. Finally, in Subsection 4.3, we describe how to reduce the size of the data set needed.

4.1 Hilbert Space Embeddings

The kernel functions mentioned in Chapter 2 can map the distribution of a random variable into an RKHS. In other words, we can represent probability distributions by elements in an RKHS with a Hilbert space embedding. In the following, assume we have probability spaces $(\mathcal{X}, \mathcal{A}, \mathcal{P}_{\mathcal{X}})$ and $(\mathcal{Y}, \mathcal{B}, \mathcal{P}_{\mathcal{Y}})$, we denote random variables by X and Y generating

samples from $\mathcal{X}$ and $\mathcal{Y}$ respectively. We endow spaces of events $\mathcal{X}$ and $\mathcal{Y}$ with corresponding σ-algebras $\mathcal{A}$ and $\mathcal{B}$ generated by Borel subsets of $\mathcal{X}$ and $\mathcal{Y}$, and denote the spaces of all probability distributions (with respect to $\mathcal{A}$ and $\mathcal{B}$) on $\mathcal{X}$ and $\mathcal{Y}$ by $\mathcal{P}_{\mathcal{X}}$ and $\mathcal{P}_{\mathcal{Y}}$, respectively.

4.1.1 Embedding Distribution [24]

Firstly, we focus on a single random variable X on sample space $\mathcal{X}$ with probability distribution P_X of random variable X. We can represent the probability distribution P_X in RKHS $\mathcal{H}$ defined by kernel k through the mapping

$$\mu : \mathcal{P}_{\mathcal{X}} \to \mathcal{H},\ P_X \to \mu_X$$

The mapping is called the kernel mean embedding mapping. The idea of kernel mean embedding is to extend the feature map φ to the space of probability distributions. The representation μ_X of the probability distribution is defined as:

$$\mu_X := \varphi(P_X) = E_X[\varphi(X)] \tag{4.1}$$

where $E_X[\cdot]$ is the expectation operator with respect to probability distribution P_X. Let $f(X)$ be a random variable which is a function of a random variable X, with $f \in \mathcal{H}$. By (2.44), we can express the expectation of $f(X)$ as the following:

$$\begin{aligned} E_X[f(X)] &= \int f(x)P_X(x)dx \\ &= \int \langle f, \varphi(x)\rangle_{\mathcal{F}} P_X(x)dx \\ &= \langle f, \int \varphi(x)P_X(x)dx\rangle_{\mathcal{F}} \\ &= \langle f, E_X[\varphi(X)]\rangle_{\mathcal{F}} \\ &= \langle f, \mu_X\rangle_{\mathcal{F}} \end{aligned} \tag{4.2}$$

As long as $E_X[k_{\mathcal{F}}(X,X)] < \infty$, μ_X will also in RKHS $\mathcal{H}$. The empirical estimate of μ_X is

$$\hat{\mu}_X = \frac{1}{n}\sum_{i=1}^{n}\varphi(x_i) \tag{4.3}$$

where $\{x_1, ..., x_n\}$ is a sample which assumed to have been drawn *i.i.d.* from P_X. The embedding μ_X guarantees that distinct distributions can be mapped to distinct points in a RKHS for the characteristic kernel class.

Definition 4.1.1. Characteristic Kernel [6] [24]: A kernel k is said to be characteristic if the mapping $\mu : \mathcal{P}_{\mathcal{X}} \to \mathcal{H}$ is injective, or equivalently, if $\forall f \in \mathcal{H},\ E_X[f(X)] = E_Y[f(Y)]$ implies $P_X = P_Y$.

There are several kinds of characteristic reproducing kernels which are popular in machine learning, in the paper by Zhu et. al [35], one of the radial based kernels: Gaussian kernel is used, which is defined as:

$$k(x,y) = \exp(-\frac{||x-y||^2}{2\sigma^2}) = \exp(-\varrho||x-y||^2) \tag{4.4}$$

where σ is variance of the Gaussian distribution and ϱ is the kernel parameter. In page 11 of the slides by Lin [20], the expression of the feature map $\varphi(x)$ is derived:

$$\begin{aligned}
\exp(\varrho||x-y||^2) =& \exp(-\varrho x^2 + 2\varrho xy - \varrho y^2) \\
=& \exp(-\varrho x^2 - \varrho y^2)(1 + \frac{2\varrho xy}{1!} + \frac{(2\varrho xy)^2}{2!} + \cdots) \\
=& \exp(-\varrho x^2 - \varrho y^2)(1 \cdot 1 + \sqrt{\frac{2\varrho}{1!}}x \cdot \sqrt{\frac{2\varrho}{1!}}y + \sqrt{\frac{(2\varrho)^2}{2!}}x^2 \cdot \sqrt{\frac{(2\varrho)^2}{2!}}y^2 + \cdots) \\
=& \varphi(x)^T\varphi(y)
\end{aligned} \tag{4.5}$$

The expression of the feature map $\varphi(x)$ is:

$$\varphi(x) = exp(-\varrho x^2)\left[1, \sqrt{\frac{2\varrho}{1!}}x, \sqrt{\frac{(2\varrho)^2}{2!}}x^2, \sqrt{\frac{(2\varrho)^3}{3!}}x^3, \cdots\right]^T \tag{4.6}$$

The dimension of the RKHS $\mathcal{H}$ associated with Gaussian kernel is infinite.

4.1.2 Cross-Covariance Operator [35]

In this part, consider two random variables X and Y with marginal probability distributions P_X and P_Y respectively, and joint probability distribution P_{XY}. Similar to $\varphi(\cdot)$ maps the random variable X into the RKHS $\mathcal{F}$, $\phi(\cdot)$ maps the random variable Y into the RKHS $\mathcal{G}$ associated with the kernel function $k_{\mathcal{G}}$. The mean map μ_Y is with respect to P_Y. Like the previous part, we can also express the expectation of $f(X)g(Y)$ as an inner product in RKHS:

$$\begin{aligned} E_{XY}[f(X)g(Y)] &= E_{XY}[\langle f, \varphi(X)\rangle_{\mathcal{F}}\langle g, \phi(Y)\rangle_{\mathcal{G}}] \\ &= E_{XY}[\langle f \otimes g, \varphi(X) \otimes \phi(Y)\rangle_{\mathcal{F}\otimes\mathcal{G}}] \\ &= \langle f \otimes g, E_{XY}[\varphi(X) \otimes \phi(Y)]\rangle_{\mathcal{F}\otimes\mathcal{G}} \end{aligned} \tag{4.7}$$

$\mathcal{F} \otimes \mathcal{G}$ is also an RKHS, where $\otimes$ is the tensor product operator:

$$V \otimes W = \begin{bmatrix} v_1w_1 & v_1w_2 & \dots & v_1w_m \\ v_2w_1 & v_2w_2 & \dots & v_2w_m \\ \vdots & \vdots & \ddots & \vdots \\ v_nw_1 & v_nw_2 & \dots & v_nw_m \end{bmatrix} \tag{4.8}$$

We define the uncentered cross-covariance operator $\mathcal{C}_{XY}$ as:

$$\begin{aligned} \mathcal{C}_{XY} &= E_{XY}[\varphi(X) \otimes \phi(Y)] \\ &= \int\int \varphi(x) \otimes \phi(y) P_{XY}(x, y)dxdy \in \mathcal{F} \otimes \mathcal{G} \end{aligned} \tag{4.9}$$

similarly, $\mathcal{C}_{XX}$ is defined as:

$$\mathcal{C}_{XX} = E[\varphi(X) \otimes \varphi(X)]$$

$$= \int \varphi(x) \otimes \varphi(x) P_X(x) dx \in \mathcal{F} \otimes \mathcal{F} \tag{4.10}$$

Then the expectation of $f(X)g(Y)$ can be expressed as:

$$\begin{aligned} E_{XY}[f(X)g(Y)] &= \langle f \otimes g, \mathcal{C}_{XY} \rangle_{\mathcal{F}\otimes\mathcal{G}} \\ &= \langle f, \mathcal{C}_{XY} g \rangle_{\mathcal{F}} \end{aligned} \tag{4.11}$$

The cross covariance operator $\mathcal{C}_{XY} : \mathcal{G} \to \mathcal{F}$ is a linear operator. Because the uncentered cross covariance operator is only determined by the joint probability distribution $P_{XY}(x, y)$ on $\mathcal{X} \times \mathcal{Y}$ when the kernel functions are given, it can be treated as joint distribution embedding μ_{XY} in the tensor product RKHS $\mathcal{F} \otimes \mathcal{G}$. [7]
The uncentered cross-covariance operator $\mathcal{C}_{XY}$ can be estimated as:

$$\begin{aligned} \hat{\mathcal{C}}_{XY} &= \frac{1}{n} \sum_{i=1}^{n} \varphi(x_i) \otimes \phi(y_i) \\ &= \frac{1}{n} \Upsilon^T \Phi \end{aligned} \tag{4.12}$$

where $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ is n pairs of training examples which are drawn $i.i.d.$ from P_{XY}, $\Upsilon = [\varphi(x_1), \cdots, \varphi(x_n)]$ and $\Phi = [\phi(y_1), \cdots, \phi(y_n)]$ are the feature matrices.

4.1.3 Conditional Embedding Operator [35] [29]

The conditional embedding operator embeds the conditional distribution $P_{Y|X}$ into a RKHS, which is introduced in [29]. Assume that the conditional expectation $E_{Y|X}[g(Y)|X = \cdot] \in \mathcal{F}$ for all $g \in \mathcal{G}$, Fukumizu's paper [5] provided the following relation:

Theorem 4.1.1. If $E[g(Y)|X = \cdot] \in \mathcal{F}$ holds for $g \in \mathcal{G}$, then

$$\mathcal{C}_{XX} E_{Y|X}[g(Y)|X = \cdot] = \mathcal{C}_{XY} g \tag{4.13}$$

If $\mathcal{C}_{XX}$ is injective, the above relation can be expressed as

$$E_{Y|X}[g(Y)|X = \cdot] = C_{XX}^{-1} C_{XY} g \tag{4.14}$$

The cross-covariance operators $\mathcal{C}_{XX}$ and $\mathcal{C}_{XY}$ map $E_{Y|X}[g(Y)|X = \cdot] \in \mathcal{F}$ and $g \in \mathcal{G}$ into $\mathcal{F}$ respectively. From (4.2), reproducing property and Theorem 4.1.1, we can express the conditional expectation of $g(Y)$ with respect to $X = x$ as an inner product:

$$\begin{aligned} E_{Y|X}[g(Y)|X = x] &= \langle g, \mu_{Y|x} \rangle_{\mathcal{G}} \\ &= \langle E_{Y|X}[g(Y)|X = \cdot], \varphi(x) \rangle_{\mathcal{F}} \\ &= \langle \mathcal{C}_{XX}^{-1} \mathcal{C}_{XY} g, \varphi(x) \rangle_{\mathcal{F}} \\ &= \langle g, (\mathcal{C}_{XX}^{-1} \mathcal{C}_{XY})^T \varphi(x) \rangle_{\mathcal{F}} \\ &= \langle g, \mathcal{C}_{YX} \mathcal{C}_{XX}^{-1} \varphi(x) \rangle_{\mathcal{G}} \end{aligned} \tag{4.15}$$

where $\mathcal{C}_{XX}$ is injective.

The term $\mu_{Y|x}$ is the conditional embedding of random variable Y given $X = x$, which is defined as:

$$\mu_{Y|x} := E_{Y|X}[\phi(Y)|X = x] \tag{4.16}$$

According to (4.15), the conditional embedding $\mu_{Y|x}$ can be expressed as

$$\mu_{Y|x} = \mathcal{C}_{YX} \mathcal{C}_{XX}^{-1} \varphi(x) \tag{4.17}$$

Note that the assumption of $E_{Y|X}[g(Y)|X = \cdot] \in \mathcal{F}$ always holds for finite domains with characteristic kernels, does not hold necessarily for continuous domains. When it does not hold, $\mathcal{C}_{YX} \mathcal{C}_{XX}^{-1} \varphi(x)$ is an approximation of $\mu_{Y|x}$ [29] [5].

By total expectation law in the RKHS, we have

$$\mu_Y = E_X[\mu_{Y|x}]$$

$$
\begin{aligned}
&= \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}E_X[\varphi(x)] \\
&= \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}\mu_X
\end{aligned} \tag{4.18}
$$

The conditional embedding operator $\mathcal{U}_{Y|X}$ is defined as following:

Definition 4.1.2. [35] The conditional embedding operator $\mathcal{U}_{Y|X}$ is defined as

$$
\mathcal{U}_{Y|X} := \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1} \tag{4.19}
$$

The conditional embedding operator $\mathcal{U}_{Y|X}$ satisfies the following three properties:

1. $\mu_{Y|x} := E_{Y|x}[\phi(Y)|X = x] = \mathcal{U}_{Y|X}k(x, \cdot) = \mathcal{U}_{Y|X}\varphi(x)$
2. $E_{Y|x}[g(Y)|X = x] = \langle g, \mu_{Y|x}\rangle_{\mathcal{G}}$
3. $\mu_Y = \mathcal{U}_{Y|X}\mu_X$

The conditional embedding operator can be estimated as

$$
\begin{aligned}
\hat{\mathcal{U}}_{Y|X} &= \hat{\mathcal{C}}_{YX}(\hat{\mathcal{C}}_{XX} + \varsigma\mathbf{I})^{-1} \\
&= \frac{1}{n}\Phi\Upsilon^T\left(\frac{1}{n}\Upsilon\Upsilon^T + \varsigma\mathbf{I}\right)^{-1} \\
&= \Phi\Upsilon^T\left(\Upsilon\Upsilon^T + \varsigma n\mathbf{I}\right)^{-1}
\end{aligned} \tag{4.20}
$$

Note that Tikhonov regularization, which alleviate the problem of near-singular matrix $\Upsilon\Upsilon^T$ for inversion by adding positive elements to the diagonals, is applied here. $\mathbf{I}$ is an $n \times n$ identity matrix, and ς is the Tikhonov regularization term. By push-through identity [11]:

$$
(\mathbf{I} + UV)^{-1}U = U(\mathbf{I} + VU)^{-1} \tag{4.21}
$$

the estimation of the conditional embedding operator can be rewritten as

$$
\hat{\mathcal{U}}_{Y|X} = \Phi(\mathbf{K} + \varsigma n\mathbf{I})^{-1}\Upsilon^T \tag{4.22}
$$

where $\mathbf{K} = \Upsilon^T\Upsilon$. The reason of the transition from (4.20) to (4.22) is because firstly, $\mathbf{K} = \Upsilon^T\Upsilon$ can be computed by the kernel trick and secondly, when we calculate $\mu_{Y|x}$ and μ_Y in (4.17) and (4.18), it is convenient to apply the estimated conditional embedding operator.

4.2 Kalman Filter in RKHS [35]

In this subsection, the Kalman filter in RKHS for multiple input multiple output nonlinear dynamical system is introduced. The estimated conditional embedding operator in Section 4.1.3 is used as the state transition operator.

As the linear mathematical system model used by the classical Kalman Filter mentioned in Section 2.1, the system in Figure 1.1 can be written as:

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{f}(\mathbf{x}_i) + \mathbf{n}_i \\ \mathbf{y}_i &= \mathbf{x}_i + \mathbf{v}_i \end{aligned} \tag{4.23}$$

where $\mathbf{x}_i$ is the state vector observed with an error, and the subscript i is the time instance, $\{\mathbf{y}_i\}$ is a measurement data set, $\mathbf{f}$ is an unknown smooth non-linear function, $\mathbf{n}_i$ and $\mathbf{v}_i$ are white noises with zero mean. We are considering a simple additive noise measurement model here.

4.2.1 Derivation of Kalman Filter in RKHS

Unlike the system models used in the classical Kalman filter or nonlinear Kalman filters, the state transition function in (4.23) is unknown. Therefore, the classical Kalman filter and nonlinear Kalman filters cannot work on the system described by (4.23). In addition, in many signal processing cases, the hidden states $\{\mathbf{x}_i\}$ are not available. Therefore we cannot use DSMCE [29] and KBR [7] algorithms, where the hidden states training data are necessary to construct the state transition operators.

To avoid the description of the hidden states dynamics which are not observable, we use the estimated embedding of the measurements $\{\mathbf{y}_i\}$ as the hidden state, instead of the embedding of $\{\mathbf{x}_i\}$. Then, the state transition operator can be constructed only using the noisy measurements. The state transition operator describe the dynamics of the measurement embedding $\mu_i = E[\varphi(\mathbf{y}_i)]$ in RKHS.

Furthermore, because the measurement embedding μ_i is not a random variable, the estimated measurement embedding $\hat{\mu}_i$, which is a random variable in RKHS is needed to serve as the hidden state in order to develop a Kalman filter in RKHS. The expectation of the random variable $E[\hat{\mu}_i]$ equals to μ_i.

Note that both $\varphi(\mathbf{y}_i)$ and $\hat{\mu}_i$ are random variables in RKHS $\mathcal{H}_\mathbf{y}$. However, $\varphi(\mathbf{y}_i)$ is the RKHS mapping of the random variable $\mathbf{y}_i$, which is in the measurement space $\mathbb{R}^{n_d}$, while for $\hat{\mu}_i$ there is no corresponding random variable in $\mathbb{R}^{n_d}$.

In the following sections, we denote the estimated conditional embedding operator $\hat{\mathcal{U}}$ as $\mathbf{F}_i$ for convenience. And $\mathbf{F}_i$ is the state transition operator instead of a matrix now. Assume we have the training set $\mathcal{D} = \{\mathbf{y}_1^0, ..., \mathbf{y}_{n+1}^0\}$, which is a block of samples taken from measurements $\{\mathbf{y}_i\}$. The data in the training set are mapped into RKHS $\mathcal{H}_\mathbf{y}$. Then we can express the estimated conditional embedding operator $\mathbf{F}_i$ as

$$\mathbf{F}_i = \Phi(\mathbf{K} + \varsigma n \mathbf{I}_n)^{-1} \Upsilon^T \tag{4.24}$$

where $\Upsilon = [\varphi(\mathbf{y}_1^0), ..., \varphi(\mathbf{y}_n^0)]$, $\Phi = [\varphi(\mathbf{y}_2^0), ..., \varphi(\mathbf{y}_{n+1}^0)]$ and $\mathbf{K} = \Upsilon^T \Upsilon$. The superscript "0" refers to the training data.

From (4.18), we know that the estimated measurement embedding in current step $\hat{\mu}_i$ can be connected to the estimated measurement embedding in next step $\hat{\mu}_{i+1}$ by

$$\hat{\mu}_{i+1} = \mathbf{F}_i \hat{\mu}_i \tag{4.25}$$

Because the operator is estimated based on limited data and no noisy dynamics, we need some noise $\boldsymbol{\xi}_i$ in RKHS added to the predicted next step embedding. The state model in

RKHS is

$$\hat{\mu}_{i+1} = \mathbf{F}_i \hat{\mu}_i + \boldsymbol{\xi}_i \tag{4.26}$$

When there is a new measurement $\mathbf{y}_i$, we have to build the measurement model in RKHS in order to link the measurement $\mathbf{y}_i$ and the estimated embedding $\hat{\mu}_i$, which is the hidden state. Then, the new measurement $\mathbf{y}_i$ is mapped into RKHS $\mathcal{H}_\mathbf{y}$ as $\varphi(\mathbf{y}_i)$. Because $\hat{\mu}_i$ is the estimate of $E[\varphi(\mathbf{y}_i)]$, the measurement model is

$$\varphi(\mathbf{y}_i) = \hat{\mu}_i + \boldsymbol{\nu}_i \tag{4.27}$$

where $\boldsymbol{\nu}_i$ is the measurement noise in RKHS. Combining (4.26) and (4.27), we have the dynamical state-space model of measurements in RKHS

$$\begin{aligned} \hat{\mu}_{i+1} &= \mathbf{F}_i \hat{\mu}_i + \boldsymbol{\xi}_i \\ \varphi(\mathbf{y}_i) &= \hat{\mu}_i + \boldsymbol{\nu}_i \end{aligned} \tag{4.28}$$

Both $\boldsymbol{\xi}_i$ and $\boldsymbol{\nu}_i$ are the noises in RKHS, which are the representations of the uncertainty in the estimated state model in (4.26) and the measurement noise in (4.27) respectively. The noises $\boldsymbol{\xi}_i$ and $\boldsymbol{\nu}_i$ are treated as $n_k \times 1$ vectors in RKHS. For the Gaussian Kernel, n_k is infinite. Assume that each component of $\boldsymbol{\xi}_i$ and $\boldsymbol{\nu}_i$ is independent zero-mean Gaussian noise process with covariance

$$\begin{aligned} \mathbf{Q}_i &= q\mathbf{I} \\ \mathbf{R}_i &= r\mathbf{I} \end{aligned} \tag{4.29}$$

Then, the Kalman filter can be used to estimate the hidden state $\hat{\mu}_i$ in RKHS. Note that with the assumptions in (4.29), the powers of the noises are infinite theoretically. The assumptions assume that the components of the noise vectors are independent of each other and have the same powers q and r. Therefore the whole powers of the noises are infinite. However, during the implementation of the Kalman filter in RKHS, the noises

are projected to the space spanned by the mappings of the training data Φ used in (4.24). Therefore, in practice, the whole powers of the noises are finite.

Although (4.29) cannot describe the covariances of the noises accurately, scalar parameters q and r can reflect the intensities of the noises. When the estimated conditional embedding operator is more accurate, a smaller q is applied. Similarly, when the measurement noise in (4.27) is larger, a larger r is applied. In addition, Zhu et. al. [35], (see pp 10-11) found that the ratio q/r is more important than the values of q and r.

Comparing the model in (2.1), which is used in classical Kalman filter with the model in (4.28), we can see that $\mathbf{F}_i = \Phi(\mathbf{K} + \varsigma n \mathbf{I}_n)^{-1}\Upsilon^T$, $\mathbf{H}_i = \mathbf{I}$, $\mathbf{Q}_i = q\mathbf{I}$ and $\mathbf{R}_i = r\mathbf{I}$. Therefore, Similar to the classical Kalman filter, the recursion to estimate μ_i in RKHS starts with $\hat{\mu}_0 = E[\varphi(\mathbf{y}_0)]$ and $\mathbf{P}_0 = \lambda\mathbf{I}$, then the following steps can be written as:

$$\hat{\mu}_i^- = \mathbf{F}_{i-1}\hat{\mu}_{i-1} \tag{4.30}$$

$$\mathbf{P}_i^- = \mathbf{F}_{i-1}\mathbf{P}_{i-1}\mathbf{F}_{i-1}^T + \mathbf{Q}_{i-1} \tag{4.31}$$

$$\mathbf{G}_i = \mathbf{P}_i^-[\mathbf{P}_i^- + \mathbf{R}_i]^{-1} \tag{4.32}$$

$$\hat{\mu}_i = \hat{\mu}_i^- + \mathbf{G}_i(\varphi(\mathbf{y}_i) - \hat{\mu}_i^-) \tag{4.33}$$

$$\mathbf{P}_i = (\mathbf{I} - \mathbf{G}_i)\mathbf{P}_i^- \tag{4.34}$$

where $\hat{\mu}_i^-$ represents the a priori estimate of the measurement embedding.

Both estimated embedding $\hat{\mu}_i$ and the mapped measurement $\varphi(\mathbf{y}_i)$ lie in RKHS $\mathcal{H}_\mathbf{y}$ with possibly infinite dimension. Therefore, $\mathbf{P}_i^-$, $\mathbf{P}_i$ and $\mathbf{G}_i$ are all operators in $\mathcal{H}_\mathbf{y} \times \mathcal{H}_\mathbf{y}$, which can be treated as $n_k \times n_k$ matrices (n_k equals to infinite for Gaussian kernel), we cannot compute the recursive approach mentioned above by doing normal matrix calculations directly. However, according to representer theorem in RKHS [27], the solution is always in the space spanned by the training data, which is dimension n in our case. Then, the problem becomes to find a way to rewrite the operators above such that $\infty \times \infty$ matrices are avoided. Inspired by [21], we can still implement the recursion approach because of the following theorems:

Theorem 4.2.1. [35] The operators $\mathbf{P}_i^-$, $\mathbf{P}_i$ and $\mathbf{G}_i$ can be calculated as the following:

$$\mathbf{P}_i^- = \Phi\tilde{\mathbf{P}}_i^-\Phi^T + q\mathbf{I} \tag{4.35}$$

$$\mathbf{P}_i = \Phi\tilde{\mathbf{P}}_i\Phi^T + \frac{qr}{q+r}\mathbf{I} \tag{4.36}$$

$$\mathbf{G}_i = \frac{r}{q+r}\Phi\tilde{\mathbf{G}}_i\Phi^T + \frac{q}{q+r}\mathbf{I} \tag{4.37}$$

where $\tilde{\mathbf{P}}_i^-$, $\tilde{\mathbf{P}}_i$ and $\tilde{\mathbf{G}}_i$ are $n \times n$ matrices. And these three matrices can be calculated recursively as the following:

$$\tilde{\mathbf{P}}_1^- = \lambda(\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\Upsilon\left[(\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\right]^T \tag{4.38}$$

$$\tilde{\mathbf{G}}_i = \left[(q+r)\mathbf{I}_n + \tilde{\mathbf{P}}_i^-\Phi^T\Phi\right]^{-1}\tilde{\mathbf{P}}_i^- \tag{4.39}$$

$$\tilde{\mathbf{P}}_i = \frac{r}{q+r}\tilde{\mathbf{P}}_i^- - \frac{r}{q+r}\tilde{\mathbf{G}}_i\Phi^T\Phi\tilde{\mathbf{P}}_i^- - \frac{qr}{q+r}\tilde{\mathbf{G}}_i \tag{4.40}$$

$$\begin{aligned}\tilde{\mathbf{P}}_{i+1}^- = &\left[(\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\right]\Phi\tilde{P}_i\Phi^T\left[(\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\right]^T \\ &+ \frac{qr}{q+r}\left[(\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\right]\left[(\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\right]^T\end{aligned} \tag{4.41}$$

Theorem 4.2.2. [35] The predicted embedding $\hat{\mu}_i^-$ and the estimated embedding $\hat{\mu}_i$ can both be computed by the mapped measurements $\Phi = \{\varphi(\mathbf{y}_i^0)\}_{i=2}^{n+1}$

$$\hat{\mu}_i^- = \Phi\mathbf{a}_i \tag{4.42}$$

$$\hat{\mu}_i = \Phi\mathbf{b}_i + \frac{q}{q+r}\varphi(\mathbf{y}_i) \tag{4.43}$$

where both $\mathbf{a}_i = [a_i, ..., a_n]^T$ and $\mathbf{b}_i = [b_1, ..., b_n]^T$ are $n \times 1$ real-valued vectors. They can be computed as the following:

$$\mathbf{a}_1 = (\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\mu_0 \tag{4.44}$$

$$\mathbf{b}_i = \left[\frac{r}{q+r}\mathbf{I} - \frac{r}{q+r}\tilde{\mathbf{G}}_i\Phi^T\Phi\right]\mathbf{a}_i + \frac{r}{q+r}\tilde{\mathbf{G}}_i\Phi^T\varphi(\mathbf{y}_i) \tag{4.45}$$

$$\mathbf{a}_i = (\mathbf{K} + \varsigma n\mathbf{I}_n)^{-1}\Upsilon^T\left[\Phi\mathbf{b}_{i-1} + \frac{q}{q+r}\varphi(\mathbf{y}_{i-1})\right] \tag{4.46}$$

Proofs of Theorem 4.2.1 and Theorem 4.2.2 are provided in Appendix. By these theorems, all of the operators can be expressed in terms of $n \times n$ matrices and $n_k \times n$ feature matrices such as Φ and Υ. Therefore, the operators can be computed only involving normal matrices calculations and inner products between feature matrices and vectors. To estimate the predicted signal $\hat{\mathbf{x}}_i$ from the estimated embedding $\hat{\mu}_i$, an approach similar to but more efficient than Pre-Image problem [3] can be used. Firstly, we have

$$\hat{\mathbf{x}}_i = E[\mathbf{x}_i] = E[\mathbf{x}_i + \mathbf{v}_i] = E[\mathbf{y}_i] \tag{4.47}$$

where $E[\mathbf{y}_i]$ can be expressed as

$$E[\mathbf{y}_i] = \left[E\left[\mathbf{y}_i^{(1)}\right], ..., E\left[\mathbf{y}_i^{(n_\mathbf{y})}\right]\right]^T = [E[f_1(\mathbf{y}_i)], ..., E[f_{n_\mathbf{y}}(\mathbf{y}_i)]]^T \tag{4.48}$$

According to (4.2), we have

$$E[f_j(\mathbf{y}_i)] = \langle f_j, \mu_i \rangle, 1 \leq j \leq n_\mathbf{y} \tag{4.49}$$

Then, the estimated input signal $\mathbf{x}_i$ is

$$\hat{\mathbf{x}}_i = [\langle f_1, \hat{\mu}_i \rangle, ..., \langle f_{n_\mathbf{y}}, \hat{\mu}_i \rangle]^T = \mathbf{f}_\mathbf{I}^T \hat{\mu}_i = \mathbf{f}_\mathbf{I}^T \hat{\varphi}(\mathbf{x}_i) \tag{4.50}$$

where $\mathbf{f}_\mathbf{I} = [f_1, ..., f_{n_\mathbf{y}}]$ is a vector consists of functions, function $f_j(\mathbf{y}) = \mathbf{y}^{(j)} (j = 1, ..., n_\mathbf{y})$, $\mathbf{y}^{(j)}$ is the jth component of vector $\mathbf{y}$. Function $f_j(\mathbf{y})$ can be approximated from the training data even though it is not in RKHS. The estimated vector of functions $\mathbf{f}_\mathbf{I}(\cdot)$ is

$$\hat{\mathbf{f}}_\mathbf{I}(\cdot) = \Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T} \tag{4.51}$$

where $\mathbf{Y}^0 = [\mathbf{y}_2^0, ..., \mathbf{y}_{n+1}^0]$. The estimated input signal can be approximated as

$$\hat{\mathbf{x}}_i = \mathbf{b}_i^T \Phi^T \Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T} + \frac{q}{q+r}\mathbf{y}_i = \mathbf{Y}^0\mathbf{b}_i + \frac{q}{q+r}\mathbf{y}_i \tag{4.52}$$

Note that the second term of (4.52) is computed from

$$\Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T}\varphi(\mathbf{y}_i) = \mathbf{y}_i \tag{4.53}$$

Now, we have the Kernel Kalman filtering based on estimated conditional embedding operator (KKF-CEO), the whole algorithm is summarized in Algorithm 1.

Algorithm 1: Kernel Kalman filtering based on estimated conditional embedding operator (KKF-CEO) [35]

Initialization: For $i = 0$, set

$\Upsilon = [\varphi(\mathbf{y}_1^0), ..., \varphi(\mathbf{y}_n^0)]$
$\Phi = [\varphi(\mathbf{y}_2^0), ..., \varphi(\mathbf{y}_{n+1}^0)]$
$\mathbf{Y}^0 = [\mathbf{y}_2^0, ..., \mathbf{y}_{n+1}^0]$
$\mathbf{K} = \Upsilon^T\Upsilon, \mathbf{M} = \Phi^T\Phi, \mathbf{T} = \Upsilon^T\Phi$
$\mathbf{L} = (\mathbf{K} + \varsigma n\mathbf{I})^{-1}$
$\mu_0 = \varphi(\mathbf{y}_0)$
$\mathbf{P}_0 = \lambda\mathbf{I}$
$\mathbf{a}_1 = \mathbf{L}\Upsilon^T\mu_0$
$\tilde{\mathbf{P}}_1^- = \lambda\mathbf{L}\mathbf{K}\mathbf{L}^T$

Filtering: For $i = 1, ...,$ compute

$$\tilde{\mathbf{G}}_i = [(q+r)\mathbf{I}_n + \tilde{\mathbf{P}}_i^-\mathbf{M}]^{-1}\tilde{\mathbf{P}}_i^-$$
$$\tilde{\mathbf{P}}_i = \frac{r}{q+r}\tilde{\mathbf{P}}_i^- - \frac{r}{q+r}\tilde{\mathbf{G}}_i\mathbf{M}\tilde{\mathbf{P}}_i^- - \frac{qr}{q+r}\tilde{\mathbf{G}}_i$$
$$\mathbf{b}_i = \left[\frac{r}{q+r}\mathbf{I}_n - \frac{r}{q+r}\tilde{\mathbf{G}}_i\mathbf{M}\right]\mathbf{a}_i + \frac{r}{q+r}\tilde{\mathbf{G}}_i\Phi^T\varphi(\mathbf{y}_i)$$
$$\hat{\mathbf{x}}_i = \mathbf{Y}^0\mathbf{b}_i + \frac{q}{q+r}\mathbf{y}_i$$
$$\mathbf{a}_{i+1} = \mathbf{L}\Upsilon^T\Phi\mathbf{b}_i + \frac{q}{q+r}\mathbf{L}\Upsilon^T\varphi(\mathbf{y}_i)$$
$$\tilde{\mathbf{P}_{i+1}^-} = \mathbf{L}\mathbf{T}\tilde{\mathbf{P}}_i\mathbf{T}^T\mathbf{L}^T + \frac{qr}{q+r}\mathbf{L}\mathbf{K}\mathbf{L}^T$$

4.3 Reducing the Size of the Training Data [35] [28] [22]

Similar to the classical Kalman filter, kernel Kalman filter is also implemented by updating the covariance matrices $\tilde{\mathbf{P}}_i^-$, $\tilde{\mathbf{P}}_i$ and $\tilde{\mathbf{G}}_i$ recursively. However, suppose we have a training set with size n to estimate the state transition operator $\mathbf{F}_i$, the size of the covariance matrices will be $n \times n$. Then the time complexity will be $O(n^3)$, and the space complexity will be $O(n^2)$. As the size of the training data increases, the time complexity will increase cubically, which is a big problem. Therefore, when the size of the training set is large, reducing the size is necessary.

Downsampling is a relatively simple and intuitive method to reduce the size of a data set. Because we assume the conditional embedding operator in KKF-CEO is estimated from *i.i.d* training data points, we can downsample the training data uniformly to reduce the time and space complexity. After downsampling with the downsampling rate L_S, the size of the training data set will be reduced to $k = \lfloor n/L_S \rfloor$, where "$\lfloor \cdot \rfloor$" is the floor operator. Then the time complexity will be reduced to $O(k^3)$ and the space complexity will be reduced to $O(k^2)$.

Quantizing is also a method to reduce the size of a data set, but it is more complicated. The authors of [35] quantize the training set by using the kernel K-means algorithm proposed by Schölkopf et al. in [28]. The algorithm is an adapted version of the K-means algorithm that can be implemented in RKHS. To understand the algorithm, we need to introduce the K-means algorithm first.

Suppose there are n observations, the propose of K-means algorithm is to partition the observations into K clusters, and each cluster has a center. Each observation belongs to the cluster with the nearest cluster center, and the sum of the distances between the data points and the cluster centers is minimum. The steps of the K-means algorithm are summarized in Algorithm 3:

Traditional K-means algorithm is implemented in Euclidean space, while the kernel K-means algorithm can be implemented in RKHS. The difference between traditional K-

Algorithm 2: K-means clustering algorithm [22]

Step 1: Specify the desired number of clusters K to be generated.

Step 2: Select K data points randomly, put each data point into a cluster, these K data points are the initial cluster centers.

Step 3: Compute the squared Euclidean distances $||\mathbf{y}_i - \mathbf{c}_j||^2$ between the data points $\mathbf{y}_i$ and the cluster centres $\mathbf{c}_j$.

Step 4: Put each data point into the cluster with the nearest cluster center.

Step 5: Compute the new center of each cluster by computing the mean of all data points in that cluster.

Step 6: Repeat Step 3 to 5 until the clusters are not changing.

means and kernel K-means is that the data points in kernel K-means are mapped into RKHS, therefore the distance used in kernel K-means algorithm is the distance in kernel space instead of Euclidean distance. The distance between $\varphi(\mathbf{y}_i)$ and $\varphi(\mathbf{y}_j)$ in kernel space is

$$\begin{aligned} ||\varphi(\mathbf{y}_i) - \varphi(\mathbf{y}_j)||^2 &= (\varphi(\mathbf{y}_i) - \varphi(\mathbf{y}_j)) \cdot (\varphi(\mathbf{y}_i) - \varphi(\mathbf{y}_j)) \\ &= \varphi(\mathbf{y}_i) \cdot \varphi(\mathbf{y}_i) + \varphi(\mathbf{y}_j) \cdot \varphi(\mathbf{y}_j) - 2\varphi(\mathbf{y}_i)\varphi(\mathbf{y}_j) \end{aligned} \tag{4.54}$$

The inner product of $\varphi(\mathbf{y}_i)$ and $\varphi(\mathbf{y}_j)$ can be computed using kernel trick $k(\mathbf{y}_i, \mathbf{y}_j) = \varphi(\mathbf{y}_i)\varphi(\mathbf{y}_j)$, we have

$$||\varphi(\mathbf{y}_i) - \varphi(\mathbf{y}_j)||^2 = k(\mathbf{y}_i, \mathbf{y}_i) + k(\mathbf{y}_j, \mathbf{y}_j) - 2k(\mathbf{y}_i, \mathbf{y}_j) \tag{4.55}$$

The equation above can be computed more efficiently by introducing a Gram matrix $\mathbf{K}$ in advance:

$$\mathbf{K} = \begin{bmatrix} k(y_1, y_1) & k(y_1, y_2) & \cdots & k(y_1, y_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(y_n, y_1) & k(y_n, y_2) & \cdots & k(y_n, y_n) \end{bmatrix} \tag{4.56}$$

Then, Equation 4.55 becomes

$$||\varphi(\mathbf{y}_i) - \varphi(\mathbf{y}_j)||^2 = \mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij} \tag{4.57}$$

Note that in [35], the authors made a modification on the gram matrix $\mathbf{K}$. The conditional embedding operators in KKF-CEO is estimated based on two mapped measurements $\Upsilon = [\varphi(\mathbf{y}_1^0), ..., \varphi(\mathbf{y}_n^0)]$ and $\Phi = [\varphi(\mathbf{y}_2^0), ..., \varphi(\mathbf{y}_{n+1}^0)]$, so they combined the mapped measurements $\varphi(\mathbf{y}_i^0)$ and $\varphi(\mathbf{y}_{i+1}^0)$ to form a new vector $v((\mathbf{y}_i^0, \mathbf{y}_{i+1}^0)) = (\varphi(\mathbf{y}_i^0)^T, \varphi(\mathbf{y}_{i+1}^0)^T) \in \mathcal{H}_\mathbf{y} \oplus \mathcal{H}_\mathbf{y}$, where $\oplus$ is the direct sum operator. The inner product between new vectors can be computed as

$$\langle v((\mathbf{y}_i^0, \mathbf{y}_{i+1}^0)), v((\mathbf{y}_j^0, \mathbf{y}_{j+1}^0))\rangle_{\mathcal{H}_\mathbf{y} \oplus \mathcal{H}_\mathbf{y}} = \langle \varphi(\mathbf{y}_i^0), \varphi(\mathbf{y}_j^0)\rangle_{\mathcal{H}_\mathbf{y}} + \langle \varphi(\mathbf{y}_{i+1}^0), \varphi(\mathbf{y}_{j+1}^0)\rangle_{\mathcal{H}_\mathbf{y}} \tag{4.58}$$

The gram matrix for the new vector $v((\mathbf{y}_i^0, \mathbf{y}_{i+1}^0))$ is denoted as $\mathbf{K}_\oplus$, which can be computed by

$$\mathbf{K}_\oplus = \mathbf{K}_1 + \mathbf{K}_2 \tag{4.59}$$

where $\mathbf{K}_1$ is the gram matrix of $\varphi(\mathbf{y}_i^0)$ and $\mathbf{K}_2$ is the gram matrix of $\varphi(\mathbf{y}_{i+1}^0)$. The gram matrix $\mathbf{K}_\oplus$ is non-negative definite and $\mathcal{H}_\mathbf{y} \oplus \mathcal{H}_\mathbf{y}$ is an RKHS.

For convenience, denote $\mathbf{y}_{i+1}^0$ as $\mathbf{z}_i^0$. After the same procedure as K-means algorithm with kernel space distance in (4.57) and modified gram matrix $\mathbf{K}$, the m pairs of training examples $\mathcal{D}_{YZ} = \{(\mathbf{y}_1^0, \mathbf{z}_1^0), ..., (\mathbf{y}_n^0, \mathbf{z}_n^0)\}$ are quantized to $\mathcal{D}_{YZ}^Q = \{(\mathbf{y}_1^0, \mathbf{z}_1^0), ..., (\mathbf{y}_k^0, \mathbf{z}_k^0)\}$. The conditional embedding operator then can be estimated using quantized training examples $\mathcal{D}_{YZ}^Q$, and the conditional embedding operator estimated by quantized training examples is denoted by $\mathbf{F}_Q$.

Theorem 4.3.1. Suppose n pairs of training examples $\mathcal{D}_{YZ} = \{(\mathbf{y}_1^0, \mathbf{z}_1^0), ..., (\mathbf{y}_n^0, \mathbf{z}_n^0)\}$ are quantized to k pairs of training examples $\mathcal{D}_{YZ}^Q = \{(\mathbf{y}_1^0, \mathbf{z}_1^0), ..., (\mathbf{y}_k^0, \mathbf{z}_k^0)\}$ using kernel K-means algorithm, then the conditional embedding operator estimated by $\mathcal{D}_{YZ}^Q$ is esti-

mated as

$$\mathbf{F}_Q = \Phi_Q \Lambda (\mathbf{K}_Q \Lambda + \varsigma n \mathbf{I}_k)^{-1} \Upsilon_Q^T \tag{4.60}$$

where $\Upsilon_Q = [\varphi(\mathbf{y}_1^q), ..., \varphi(\mathbf{y}_k^q)]$ and $\Phi_Q = [\varphi(\mathbf{z}_1^q), ..., \varphi(\mathbf{z}_k^q)]$, $\mathbf{K}_Q = \Upsilon_Q^T \Upsilon_Q^T$, $\Lambda = diag[\lambda_1^q, ..., \lambda_k^1]$, the entries $\lambda_j^q, 1 \leq j \leq k$ are the number of training examples in the jth cluster. The proof of Theorem 4.3.1 is provided in Appendix.

Also, from Theorem 4.3.1, we can get that the matrix $\mathbf{L}$ in KKF-CEO is computed as

$$\mathbf{L} = \Lambda (\mathbf{K} + \varsigma n \mathbf{I})^{-1} \tag{4.61}$$

Chapter 5

Double Sided Kernel for Nonlinear Systems

In this chapter, we are introducing a novel algorithm that is adapting the double sided kernel (KDS) mentioned in Chapter 3 for SISO nonlinear systems such as the system in (6.4). The novel algorithm estimates the true outputs $\{x(t)\}$ from noisy measurements $\{y(t)\}$ within arbitrary interval of time $t \in [a, b]$. The double sided kernel for nonlinear models also employs forward integration and backward integration.

5.1 Derivation of KDS for Nonlinear System

Inspired by KKF-CEO, we embed the true outputs $\{x(t)\}$ and noisy measurements $\{y(t)\}$ into RKHS $\mathcal{H}_{\mathbf{x}}$ and $\mathcal{H}_{\mathbf{y}}$ respectively using the Gaussian feature map $\varphi(\cdot)$ from (4.6) in our novel algorithm, then we use the embedding of noisy measurements to estimate the true outputs $\{x(t)\}$. The embedding of true outputs $\{x(t)\}$ and noisy measurements $\{y(t)\}$ are vectors with infinite dimensions which are denoted as

$$\varphi_{\mathbf{x}}(t) = \varphi(x(t)) = exp(-\varrho x(t)^2) \left[1, \sqrt{\frac{2\varrho}{1!}}x(t), \sqrt{\frac{(2\varrho)^2}{2!}}x(t)^2, \sqrt{\frac{(2\varrho)^3}{3!}}x(t)^3, \cdots\right]^T \tag{5.1}$$

$$\varphi_{\mathbf{y}}(t) = \varphi(y(t)) = exp(-\varrho y(t)^2)\left[1, \sqrt{\frac{2\varrho}{1!}}y(t), \sqrt{\frac{(2\varrho)^2}{2!}}y(t)^2, \sqrt{\frac{(2\varrho)^3}{3!}}y(t)^3, \cdots\right]^T \tag{5.2}$$

respectively [20]. On time interval $t \in [a, b]$, we adopt the following local approximation of the true unknown nonlinear system model by infinitely dimensional model in the transformed embedding $\varphi_{\mathbf{x}}(t)$

$$\varphi_{\mathbf{x}}^{(1)}(t) + \mathbf{A}\varphi_{\mathbf{x}}(t) = \mathbf{0} \tag{5.3}$$

where $\mathbf{A}$ is an infinite matrix. Note that a higher order approximation use instead of (5.3) can provide a fit for original nonlinear system trajectories.

The equation in (5.3) is in a similar form to the equation in (3.8) that is used in the linear case, therefore, we can derive the double sided kernel for nonlinear systems using a similar procedure to linear system case.

Multiply (5.3) by $(\epsilon - a)$ and $(b - \zeta)$, we have

$$(\epsilon - a)\varphi_{\mathbf{x}}^{(1)}(t) + (\epsilon - a)\mathbf{A}\varphi_{\mathbf{x}}(t) = \mathbf{0} \tag{5.4}$$

$$(b - \zeta)\varphi_{\mathbf{x}}^{(1)}(t) + (b - \zeta)\mathbf{A}\varphi_{\mathbf{x}}(t) = \mathbf{0} \tag{5.5}$$

Integrate the (5.4) in the forward direction on interval $[a, a + \tau]$, we have

$$\begin{aligned}
&\int_a^{a+\tau} (\epsilon - a)\varphi_{\mathbf{x}}^{(1)}(\epsilon)d\epsilon + \int_a^{a+\tau} (\epsilon - a)\mathbf{A}\varphi_{\mathbf{x}}(\epsilon)\mathrm{d}\epsilon \\
&\quad = (\epsilon - a)\varphi_{\mathbf{x}}(\epsilon)|_a^{a+\tau} + \int_a^{a+\tau} ((\epsilon - a)\mathbf{A} - \mathbf{I})\varphi_{\mathbf{x}}(\epsilon)\mathrm{d}\epsilon \\
&\quad = \tau\varphi_{\mathbf{x}}(a + \tau) + \int_a^{a+\tau} ((\epsilon - a)\mathbf{A} - \mathbf{I})\varphi_{\mathbf{x}}(\epsilon)\mathrm{d}\epsilon
\end{aligned} \tag{5.6}$$

where I is the infinity identity matrix. Let $a + \tau = t$ and apply Cauchy's formula to (5.6),

$$(t - a)\varphi_{\mathbf{x}}(t) \triangleq \int_a^t K_F(t, \tau)\varphi_{\mathbf{x}}(\tau)\mathrm{d}\tau \tag{5.7}$$

in which $K_F(t,\tau)$ is

$$K_F(t,\tau) = \mathbf{I} - (\tau - a)\mathbf{A} \tag{5.8}$$

Then, integrate (5.5) in the backward direction on interval $[b-\sigma, b]$, we have

$$\begin{aligned}
&\int_{b-\sigma}^{b} (b-\zeta)\varphi_{\mathbf{x}}^{(1)}(\zeta)\mathrm{d}\zeta + \int_{b-\sigma}^{b} (b-\zeta)\mathbf{A}\varphi_{\mathbf{x}}(\zeta)\mathrm{d}\zeta \\
&\quad = (b-\zeta)\varphi_{\mathbf{x}}(\zeta)|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} ((b-\zeta)\mathbf{A} - \mathbf{I})\varphi_{\mathbf{x}}(\zeta)\mathrm{d}\zeta \\
&\quad = -\sigma\varphi_{\mathbf{x}}(b-\sigma) + \int_{b-\sigma}^{b} ((b-\zeta)\mathbf{A} - \mathbf{I})\varphi_{\mathbf{x}}(\zeta)\mathrm{d}\zeta
\end{aligned} \tag{5.9}$$

Let $b - \sigma = t$ and apply Cauchy's formula, we have

$$(b-t)\varphi_{\mathbf{x}}(t) \triangleq \int_{t}^{b} K_B(t,\tau)\varphi_{\mathbf{x}}(\tau)\mathrm{d}\tau \tag{5.10}$$

in which $K_B(t,\tau)$ is

$$K_B(t,\tau) = \mathbf{I} + (b-\tau)\mathbf{A} \tag{5.11}$$

Add (5.7) and (5.10) and divide both sides by $b-a$, we have

$$\varphi_{\mathbf{x}}(t) = \int_{a}^{b} K_{DS}(t,\tau)\varphi_{\mathbf{x}}(\tau)\mathrm{d}\tau \tag{5.12}$$

where

$$K_{DS} \triangleq \frac{1}{b-a}\begin{cases} \mathbf{I} - (\tau - a)\mathbf{A} : \tau \le t \\ \mathbf{I} + (b-\tau)\mathbf{A} : \tau > t \end{cases} \tag{5.13}$$

5.2 State Estimation of Nonlinear Systems Using Double Sided Kernel

Considering the system in (5.3), we need first to estimate the infinite matrix $\mathbf{A}$ in order to perform state estimation. Suppose there are finite number n of noisy observations $\{y(t_i)\}$ correspond to time instances $\{t_i\}$, where $i = 1, ..., n$. We need to determine the matrix $\mathbf{A}$ by finding the optimal solution of cost function $J(\mathbf{A})$ similar to (3.64):

$$\min\{J(\mathbf{A}) := \frac{1}{2n}\sum_{i=1}^{n}\left\|\varphi_{\mathbf{y}}(t_i) - \int_a^b K_{DS}(t_i,\tau)\varphi_{\mathbf{y}}(\tau)d\tau\right\|^2 \mid \text{w.r.t. } \mathbf{A} \in \{\mathbf{A}\}\} \tag{5.14}$$

where $\{\mathbf{A}\}$ is the set of all infinite by infinite real matrices. Express the double sided kernel as the sum of scalar product of matrix $\mathbf{A}$ and a partial kernel similar to (3.65):

$$K_{DS}(t,\tau) = K_v(t,\tau)\mathbf{A} + K_{v2}\mathbf{I} \tag{5.15}$$

where

$$K_v(t,\tau) = \frac{1}{b-a}\begin{cases} -(\tau - a) : \tau \le t \\ (b-\tau) : \tau > t \end{cases} \tag{5.16}$$

$$K_{v2} = \frac{1}{b-a} \tag{5.17}$$

We now rewrite the cost function in (5.20) as follows

$$\begin{aligned} J(\mathbf{A}) :=& \frac{1}{2n}\sum_{i=1}^{n}\left\|\varphi_{\mathbf{y}}(t_i) - \int_a^b K_{DS}(t_i,\tau)\,\varphi_{\mathbf{y}}(\tau)d\tau\right\|^2 \\ =& \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{2}\left\|\varphi_{\mathbf{y}}(t_i)\right\|^2 - \varphi_{\mathbf{y}}(t_i)^T\int_a^b K_{DS}(t_i,\tau)\,\varphi_{\mathbf{y}}(\tau)d\tau + \frac{1}{2}\left\|\int_a^b K_{DS}(t_i,\tau)\,\varphi_{\mathbf{y}}(\tau)d\tau\right\|^2\right] \\ =& \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{2}\left\|\varphi_{\mathbf{y}}(t_i)^2\right\| - \varphi_{\mathbf{y}}(t_i)^T\int_a^b K_{DS}(t_i,\tau)\,\varphi_{\mathbf{y}}(\tau)d\tau\right. \end{aligned}$$

$$
\begin{aligned}
&+\frac{1}{2}\left[\int_a^b K_{DS}(t_i,\tau)\ \varphi_{\mathbf{y}}(\tau)d\tau\right]^T\int_a^b K_{DS}(t_i,s)\ \varphi_{\mathbf{y}}(s)ds\Big]\\
=&\frac{1}{2n}\sum_{i=1}^{n}\left\|\varphi_{\mathbf{y}}(t_i)\right\|^2-\frac{1}{n}\sum_{i=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\int_a^b K_{DS}(t_i,\tau)\ \varphi_{\mathbf{y}}(\tau)\ d\tau\\
&+\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b\left[K_{DS}(t_i,\tau)\varphi_{\mathbf{y}}(\tau)\right]^T K_{DS}(t_i,s)\ \varphi_{\mathbf{y}}(s)\ d\tau\ ds
\end{aligned}
$$

Then substitute double sided kernel with the right side of (5.15), obtaining (5.18)

$$
\begin{aligned}
J(\mathbf{A})=&\frac{1}{2n}\sum_{i=1}^{n}\left\|\varphi_{\mathbf{y}}(t_i)\right\|^2-\frac{1}{n}\sum_{i=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\int_a^b\left[K_v(t_i,\tau)\mathbf{A}+K_{v2}\mathbf{I}\right]\varphi_{\mathbf{y}}(\tau)\ d\tau\\
&+\frac{1}{2n}\sum_{i=1}^{n}\int_a^b\int_a^b\varphi_{\mathbf{y}}(\tau)^T[K_v(t_i,\tau)\mathbf{A}+K_{v2}\mathbf{I}]^T[K_v(t_i,s)\mathbf{A}+K_{v2}\mathbf{I}]\ \varphi_{\mathbf{y}}(s)\ d\tau\ ds
\end{aligned}
\tag{5.18}
$$

Replacing the integral in (5.18) with their finite sum approximation gives

$$
\begin{aligned}
J(\mathbf{A})=&\frac{1}{2n}\sum_{i=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_i)-\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\left[K_v(t_i,t_j)\mathbf{A}+K_{v2}\mathbf{I}\right]\varphi_{\mathbf{y}}(t_j)\Delta t\\
&+\frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\varphi_{\mathbf{y}}(t_j)^T\left[K_v(t_i,t_j)\mathbf{A}+K_{v2}\mathbf{I}\right]^T\left[K_v(t_i,t_k)\mathbf{A}+K_{v2}\mathbf{I}\right]\varphi_{\mathbf{y}}(t_k)\Delta t^2\\
=&\frac{1}{2n}\sum_{i=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_i)-\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\left[K_v(t_i,t_j)\mathbf{A}\varphi_{\mathbf{y}}(t_j)+K_{v2}\varphi_{\mathbf{y}}(t_j)\right]\Delta t\\
&+\frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[K_v(t_i,t_j)\mathbf{A}\varphi_{\mathbf{y}}(t_j)+K_{v2}\varphi_{\mathbf{y}}(t_j)\right]^T\left[K_v(t_i,t_k)\mathbf{A}\varphi_{\mathbf{y}}(t_k)+K_{v2}\varphi_{\mathbf{y}}(t_k)\right]\Delta t^2
\end{aligned}
\tag{5.19}
$$

where Δt is the time step of noisy observations, $\Delta t = t_{i+1} - t_i$.

It is impossible to solve the entire infinite matrix $\mathbf{A} \in \{\mathbf{A}\}$ such that $J(\mathbf{A})$ is the minimum. But if we only consider the set of infinite matrices $\{\tilde{\mathbf{A}}\} \subset \{\mathbf{A}\}$ that have $\{\varphi_{\mathbf{y}}(t_i)\}$ as their eigenvectors and find the optimal solution of cost function $J(\tilde{\mathbf{A}})$ instead:

$$
\min\{J(\tilde{\mathbf{A}}) := \frac{1}{2n}\sum_{i=1}^{n}\left\|\varphi_{\mathbf{y}}(t_i)-\int_a^b K_{DS}(t_i,\tau)\varphi_{\mathbf{y}}(\tau)d\tau\right\|^2 \mid \text{w.r.t. } \tilde{\mathbf{A}}\in\{\tilde{\mathbf{A}}\}\}
\tag{5.20}
$$

Then at each time instance, we have

$$\tilde{\mathbf{A}}\varphi_{\mathbf{y}}(t_i) = a_i\varphi_{\mathbf{y}}(t_i) \tag{5.21}$$

where a_i is the eigenvalue of matrix $\tilde{\mathbf{A}}$ corresponds to eigenvector $\varphi_{\mathbf{y}}(t_i)$. By (5.21), we replace the infinite matrix $\tilde{\mathbf{A}}$ with eigenvalues $\{a_i\}$ of it correspond to eigenvectors $\{\varphi_{\mathbf{y}}(t_i)\}$. Hence we only need to find the eigenvalues $\{a_i\}$ that can minimize cost function $J(\tilde{\mathbf{A}})$ instead of the entire infinite matrix $\tilde{\mathbf{A}}$. Bring (5.21) into (5.19), we have

$$\begin{aligned}
J(\tilde{\mathbf{A}}) =& \frac{1}{2n}\sum_{i=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_i) - \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\left[K_v(t_i,t_j)a_j\varphi_{\mathbf{y}}(t_j) + K_{v2}\varphi_{\mathbf{y}}(t_j)\right]\Delta t \\
&+ \frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[K_v(t_i,t_j)a_j\varphi_{\mathbf{y}}(t_j) + K_{v2}\varphi_{\mathbf{y}}(t_j)\right]^T\left[K_v(t_i,t_k)a_k\varphi_{\mathbf{y}}(t_k) + K_{v2}\varphi_{\mathbf{y}}(t_k)\right]\Delta t^2 \\
=& \frac{1}{2n}\sum_{i=1}^{n}\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_i) - \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[K_v(t_i,t_j)a_j\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_j) + K_{v2}\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_j)\right]\Delta t \\
&+ \frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\Big[a_ja_kK_v(t_i,t_j)K_v(t_i,t_k)\varphi_{\mathbf{y}}(t_j)^T\varphi_{\mathbf{y}}(t_k) + a_jK_v(t_i,t_j)K_{v2}\varphi_{\mathbf{y}}(t_j)^T\varphi_{\mathbf{y}}(t_k) \\
&\qquad + a_kK_v(t_i,t_k)K_{v2}\varphi_{\mathbf{y}}(t_j)^T\varphi_{\mathbf{y}}(t_k) + K_{v2}^2\varphi_{\mathbf{y}}(t_j)^T\varphi_{\mathbf{y}}(t_k)\Big]\Delta t^2
\end{aligned} \tag{5.22}$$

By kernel trick (4.4), we have

$$\varphi_{\mathbf{y}}(t_i)^T\varphi_{\mathbf{y}}(t_j) = K_G(y_i,y_j) = exp(-\varrho||y(t_i) - y(t_j)||^2) \tag{5.23}$$

Here we denote $y(t_i)$ as y_i for convenience. Then we have

$$\begin{aligned}
J(\tilde{\mathbf{A}}) =& \frac{1}{2n}\sum_{i=1}^{n}K_G(y_i,y_i) - \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[K_v(t_i,t_j)a_jK_G(y_i,y_j) + K_{v2}K_G(y_i,y_j)\right]\Delta t \\
&+ \frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\Big[a_ja_kK_v(t_i,t_j)K_v(t_i,t_k)K_G(y_j,y_k) \\
&\qquad + a_jK_v(t_i,t_j)K_{v2}K_G(y_j,y_k) + a_kK_v(t_i,t_k)K_{v2}K_G(y_j,y_k) \\
&\qquad + K_{v2}^2K_G(y_j,y_k)\Big]\Delta t^2
\end{aligned}$$

$$
\begin{aligned}
=&\frac{1}{2n}\sum_{i=1}^{n}\left[K_G(y_i,y_i)-\sum_{j=1}^{n}2K_{v2}K_G(y_i,y_j)\Delta t+\sum_{j=1}^{n}\sum_{k=1}^{n}K_{v2}^2K_G(y_j,y_k)\Delta t^2\right]\\
&+\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[-K_v(t_i,t_j)K_G(y_i,y_j)\Delta t+\sum_{k=1}^{n}K_v(t_i,t_j)K_{v2}K_G(y_j,y_k)\Delta t^2\right]a_j\\
&+\frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[a_ja_kK_v(t_i,t_j)K_v(t_i,t_k)K_G(y_j,y_k)\right]\Delta t^2
\end{aligned}
\tag{5.24}
$$

The equation above can be expressed as standard quadratic form,

$$
J(\mathbf{a})=d+\mathbf{b}^T\mathbf{a}+\frac{1}{2}\mathbf{a}^T\mathbf{C}\mathbf{a}
\tag{5.25}
$$

where d is a constant:

$$
\begin{aligned}
d=&\frac{1}{2n}\sum_{i=1}^{n}\Bigg[K_G(y_i,y_i)\\
&-\sum_{j=1}^{n}2K_{v2}K_G(y_i,y_j)\Delta t+\sum_{j=1}^{n}\sum_{k=1}^{n}K_{v2}^2K_G(y_j,y_k)\Delta t^2\Bigg]
\end{aligned}
\tag{5.26}
$$

$\mathbf{b}$ is a vector with n entries:

$$
\mathbf{b}=\frac{\Delta t^2K_{v2}}{n}\begin{bmatrix}\sum_{i=1}^{n}K_v(t_i,t_1)\sum_{k=1}^{n}K_G(y_k,y_1))\\ \vdots\\ \sum_{i=1}^{n}K_v(t_i,t_n)\sum_{k=1}^{n}K_G(y_k,y_n)\end{bmatrix}-\frac{\Delta t}{n}\begin{bmatrix}\sum_{i=1}^{n}K_v(t_i,t_1)K_G(y_i,y_1)\\ \vdots\\ \sum_{i=1}^{n}K_v(t_i,t_n)K_G(y_i,y_n)\end{bmatrix}
\tag{5.27}
$$

$\mathbf{C}$ is a $n\times n$ matrix:

$$
\mathbf{C}=\frac{1}{n}\begin{bmatrix}\sum_{i=1}^{n}K_v(t_i,t_1)K_v(t_i,t_1)K_G(y_1,y_1) & \cdots & \sum_{i=1}^{n}K_v(t_i,t_1)K_v(t_i,t_n)K_G(y_1,y_n)\\ \vdots & \ddots & \vdots\\ \sum_{i=1}^{n}K_v(t_i,t_n)K_v(t_i,t_1)K_G(y_n,y_1) & \cdots & \sum_{i=1}^{n}K_v(t_i,t_n)K_v(t_i,t_n)K_G(y_n,y_n)\end{bmatrix}
\tag{5.28}
$$

And $\mathbf{a}$ is a vector consists of eigenvalues of $\hat{\mathbf{A}}$:

$$\mathbf{a} = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}^T \tag{5.29}$$

And $\min\{J(\mathbf{a})|\mathbf{a} \in \mathcal{R}^n\}$ can be attained globally and uniquely at

$$\hat{\mathbf{a}} = \mathbf{C}^{-1}\mathbf{b} \tag{5.30}$$

The reconstruction step in nonlinear double sided kernel is similar to (3.71) in linear case, while the noisy measures in linear case are changed into the embedded noisy measurements, and the estimate of the outputs in linear case are changed into the estimates of the embedded outputs here. The estimate of the embedded output at time t_i is

$$\hat{\varphi}_{\mathbf{x}}(t_i) = \int_a^b K_{DS}(t_i, \tau)\varphi_{\mathbf{y}}(\tau)d\tau \tag{5.31}$$

Discretize the integral above, we have

$$\begin{aligned} \hat{\varphi}_{\mathbf{x}}(t_i) &= \sum_{i=1}^{n} K_{DS}(t_i, t_j)\varphi_{\mathbf{y}}(t_j)\Delta t \\ &= \frac{\Delta t}{b-a} \begin{cases} \sum_{i=1}^{n} \varphi_{\mathbf{y}}(t_j) - (t_j - a)\hat{\mathbf{A}}\varphi_{\mathbf{y}}(t_j) : t_j \leq t_i \\ \sum_{i=1}^{n} \varphi_{\mathbf{y}}(t_j) + (b - t_j)\hat{\mathbf{A}}\varphi_{\mathbf{y}}(t_j) : t_j > t_i \end{cases} \\ &= \frac{\Delta t}{b-a} \begin{cases} \sum_{i=1}^{n} \varphi_{\mathbf{y}}(t_j) - (t_j - a)\hat{a}_j\varphi_{\mathbf{y}}(t_j) : t_j \leq t_i \\ \sum_{i=1}^{n} \varphi_{\mathbf{y}}(t_j) + (b - t_j)\hat{a}_j\varphi_{\mathbf{y}}(t_j) : t_j > t_i \end{cases} \end{aligned} \tag{5.32}$$

By (4.50) which is used by Zhu et. al [35], we can get the estimated outputs $\{\hat{x}(t_i)\}$:

$$\hat{x}(t_i) = \mathbf{f}_{\mathbf{I}}^T\hat{\varphi}_{\mathbf{x}}(t_i) = \Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T}\hat{\varphi}_{\mathbf{x}}(t_i) \tag{5.33}$$

Multiply both sides of (5.32) by $\mathbf{f}_\mathbf{I}^T$, we have

$$\begin{aligned}\hat{x}(t_i) &= \frac{\Delta t}{b-a}\begin{cases}\sum_{i=1}^{n}\Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T}\varphi_\mathbf{y}(t_j) - (t_j-a)a_j\Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T}\varphi_\mathbf{y}(t_j) : t_j \le t_i \\ \sum_{i=1}^{n}\Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T}\varphi_\mathbf{y}(t_j) + (b-t_j)a_j\Phi(\Phi^T\Phi)^{-1}\mathbf{Y}^{0^T}\varphi_\mathbf{y}(t_j) : t_j > t_i\end{cases} \\ &= \frac{\Delta t}{b-a}\begin{cases}\sum_{i=1}^{n} y(t_j) - (t_j-a)a_j y(t_j) : t_j \le t_i \\ \sum_{i=1}^{n} y(t_j) + (b-t_j)a_j y(t_j) : t_j > t_i\end{cases}\end{aligned} \tag{5.34}$$

Our novel algorithm is summarized in Algorithm 3:

Algorithm 3: Adapted double sided kernel for nonlinear systems (NLKDS)

Step 1: Compute matrices **M** and **G**, where

$$\mathbf{M}_{i,j} = \frac{1}{b-a}\begin{cases}-(t_j-a) : t_j \le t_i \\ (b-t_j) : t_j > t_i\end{cases}, \mathbf{G}_{i,j} = exp(-\varrho||y(t_i)-y(t_j)||^2)$$

Step 2: Compute vector **b**,

$$\mathbf{b} = \frac{\Delta t^2 K_{v2}}{n}\begin{bmatrix}\sum_{i=1}^{n}\mathbf{M}_{i,1}\sum_{k=1}^{n}\mathbf{G}_{k,1} \\ \vdots \\ \sum_{i=1}^{n}\mathbf{M}_{i,n}\sum_{k=1}^{n}\mathbf{G}_{k,n}\end{bmatrix} - \frac{\Delta t}{n}\begin{bmatrix}\sum_{i=1}^{n}\mathbf{M}_{i,1}\mathbf{G}_{i,1} \\ \vdots \\ \sum_{i=1}^{n}\mathbf{M}_{i,n}\mathbf{G}_{i,n}\end{bmatrix}$$

Step 3: Compute matrix **C**,

$$\mathbf{C} = \frac{1}{n}\begin{bmatrix}\sum_{i=1}^{n}\mathbf{M}_{i,1}\mathbf{M}_{i,1}\mathbf{G}_{1,1} & \cdots & \sum_{i=1}^{n}\mathbf{M}_{i,1}\mathbf{M}_{i,n}\mathbf{G}_{1,n} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{n}\mathbf{M}_{i,n}\mathbf{M}_{i,1}\mathbf{G}_{n,1} & \cdots & \sum_{i=1}^{n}\mathbf{M}_{i,n}\mathbf{M}_{i,n}\mathbf{G}_{n,n}\end{bmatrix}$$

Step 4: Estimate **a**,

$$\hat{\mathbf{a}} = \mathbf{C}^{-1}\mathbf{b}$$

Step 5: The estimated true output time series $\{\hat{x}(t_i)\}$ is

$$\hat{x}(t_i) = \frac{\Delta t}{b-a}\begin{cases}\sum_{i=1}^{n} y(t_j) - (t_j-a)\hat{\mathbf{a}}_j y(t_j) : t_j \le t_i \\ \sum_{i=1}^{n} y(t_j) + (b-t_j)\hat{\mathbf{a}}_j y(t_j) : t_j > t_i\end{cases}$$

Note that the time complexity of computing $\sum_{i=1}^{n} \mathbf{M}_{i,j}\mathbf{M}_{i,k}$ during computing $\mathbf{C}$ is $O(n)$, Therefore, the time complexity of the whole algorithm is $O(n^3)$. However, matrix $\mathbf{M}$ is in the following format:

$$\mathbf{M} = \begin{bmatrix} b_1 & a_2 & a_3 & a_4 & \cdots & a_{n-1} & a_n \\ b_1 & b_2 & a_3 & a_4 & \cdots & a_{n-1} & a_n \\ b_1 & b_2 & b_3 & a_4 & \cdots & a_{n-1} & a_n \\ b_1 & b_2 & b_3 & b_4 & \cdots & a_{n-1} & a_n \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_1 & b_2 & b_3 & b_4 & \cdots & b_{n-1} & a_n \end{bmatrix} \tag{5.35}$$

Which means we can simplify the computation of $\sum_{i=1}^{n} \mathbf{M}_{i,j}\mathbf{M}_{i,k}$ to

$$\begin{gathered} \sum_{i=1}^{n} \mathbf{M}_{i,j}\mathbf{M}_{i,k} = (s-1)\mathbf{M}_{1,j}\mathbf{M}_{1,k} + (l-s)\mathbf{M}_{s,j}\mathbf{M}_{s,k} + (n-l+1)\mathbf{M}_{l,j}\mathbf{M}_{l,k} \\ s = \min(j,k), l = \max(j,k) \end{gathered} \tag{5.36}$$

The time complexity of computing $\sum_{i=1}^{n} \mathbf{M}_{i,j}\mathbf{M}_{i,k}$ is reduced to $O(1)$, hence the time complexity of the whole algorithm is reduced to $O(n^2)$.

Chapter 6

Experiments and Results

In this chapter, we first compare the performances of the linear double sided kernel (linear KDS), the kernel Kalman filter (KKF-CEO) and our novel nonlinear double sided kernel (NLKDS) for fourth order linear system under several noise levels. Then we compare the performances of the KKF-CEO and our NLKDS for Van Der Pol equation, Sedoglavic equation and IKEDA chaotic dynamical system under various noise levels. We are comparing the performances of algorithms based on accuracy and running time. Both mean squares error (MSE) and signal to noise ratio (SNR) of estimated outputs are used to measure the accuracy of algorithms, where MSE is calculated by

$$MSE = \frac{1}{n_{data}} \sum_{i=1}^{n_{data}} ||x_i - \hat{x}_i||^2 \tag{6.1}$$

and SNR is calculated by

$$SNR = 10\log_{10}(\frac{P_{signal}}{P_{noise}}) \tag{6.2}$$

6.1 Performances on Linear Systems

First, we consider a fourth order linear system with the following characteristic equation

$$x^{(4)}(t) + 10x^{(2)}(t) + 10x^{(1)}(t) + x(t) = 0 \tag{6.3}$$

The initial condition $\begin{bmatrix} x^{(3)}(0) & x^{(2)}(0) & x^{(1)}(0) & x(0) \end{bmatrix}$ is $\begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$. The system is an unstable system, which is difficult to estimate. The noisy measurements $\{y(t_i)\}$ are generated by adding white Gaussian noise (AWGN) with mean $\mu = 0$ and $SNR = \{0, 10, 20\}dB$ to the true outputs $\{x(t_i)\}$. The time interval $[a, b]$ of estimation in this section is $[0, 6]$ while the number of sample points is different in each algorithm. The variance of the Gaussian distribution σ in KKF-CEO and nonlinear KDS is set to the median of the pairwise distance of noisy measurements, the values of q are tuned to 1 and the values of r are tuned to $\{0.5, 1.5, 6\}$ corresponding to $SNR = \{20, 10, 0\}dB$, ζ and λ are set to 10^{-3} and 10^{-4} in KKF-CEO Figure 6.1-6.9 are the plots of the noisy measurements, the true outputs and the outputs estimated by the compared algorithms under different noise levels. Table 6.1 shows the performances of compared algorithms under different noise levels. From the results, we can see that for linear dynamical systems, our novel algorithm is more accurate than Zhu's algorithm, while linear KDS is more accurate than our novel algorithm. However, the runtime of our algorithm is much shorter than the other two algorithms.

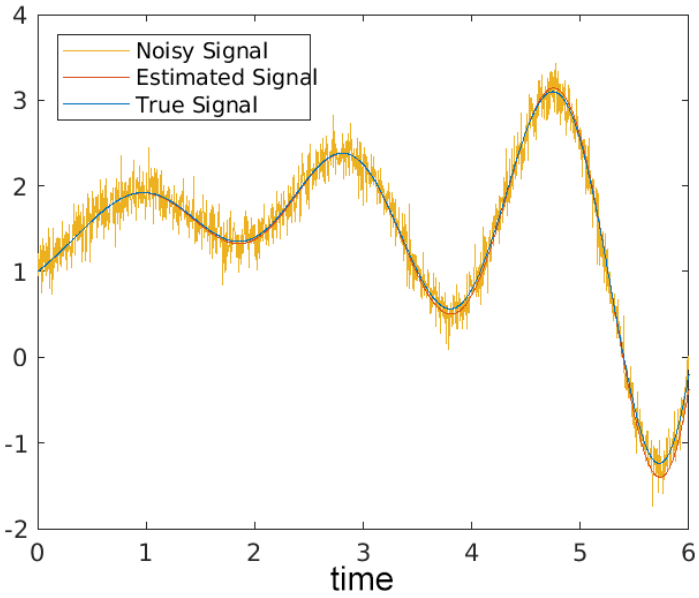

Figure 6.1: Noisy, true and estimated 4th order linear system outputs by linear KDS with AWGN of $\mu = 0$, SNR=20dB and N=1500.

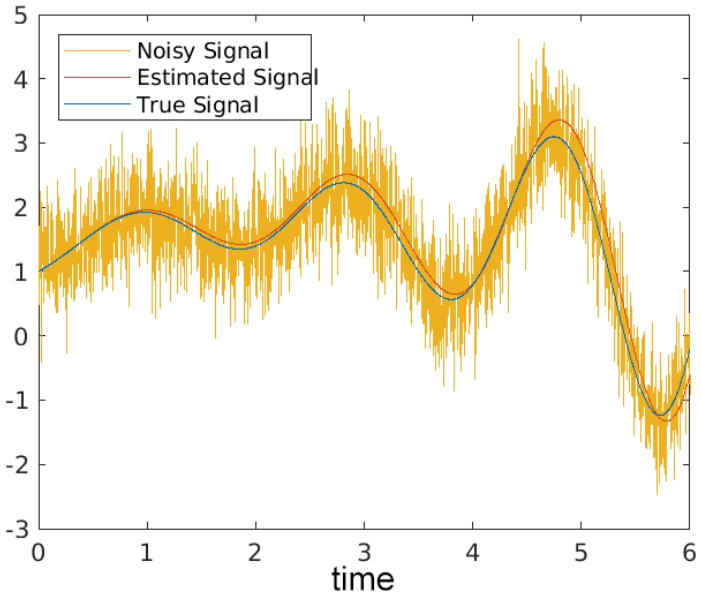

Figure 6.2: Noisy, true and estimated 4th order linear system outputs by linear KDS with AWGN of $\mu = 0$, SNR=10dB and N=2400.

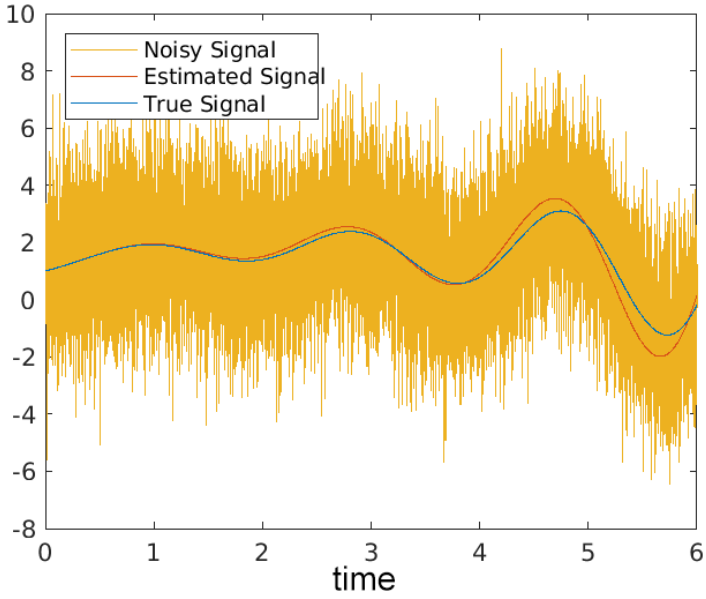

Figure 6.3: Noisy, true and estimated 4th order linear system outputs by linear KDS with AWGN of $\mu = 0$, SNR=0dB and N=15000.

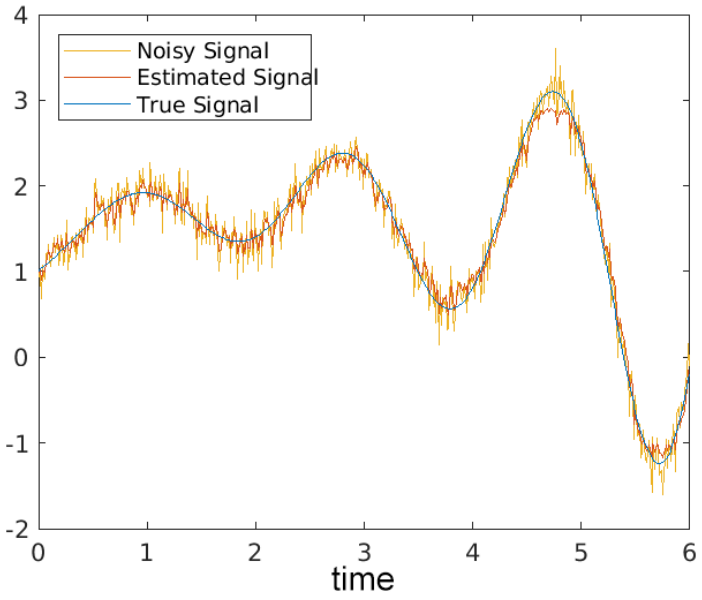

Figure 6.4: Noisy, true and estimated 4th order linear system outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=20dB and N=600.

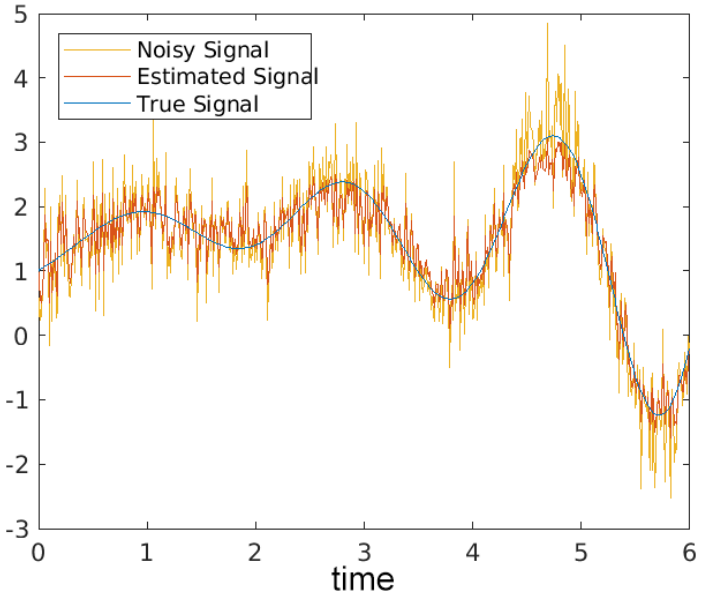

Figure 6.5: Noisy, true and estimated 4th order linear system outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=10dB and N=600.

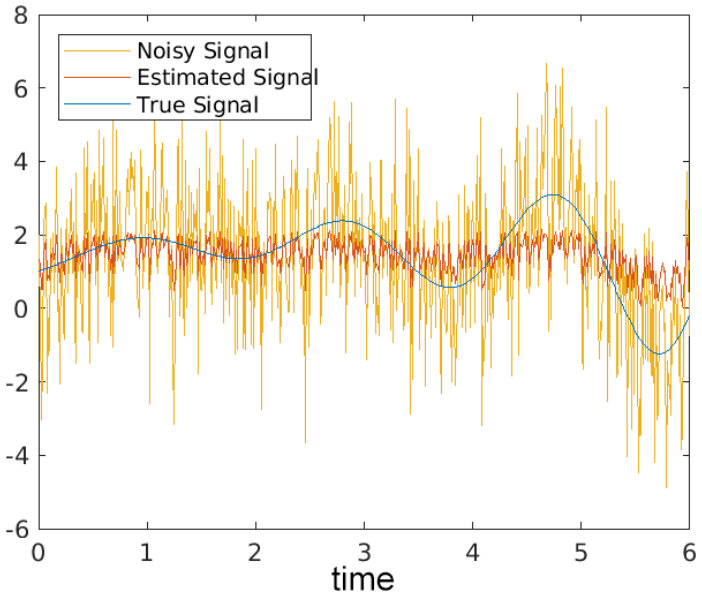

Figure 6.6: Noisy, true and estimated 4th order linear system outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=0dB and N=600.

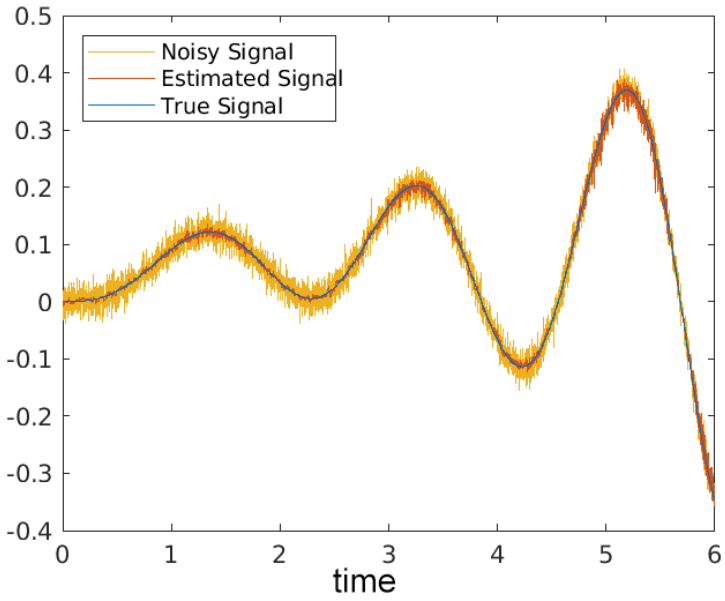

Figure 6.7: Noisy, true and estimated 4th order linear system outputs by NLKDS with AWGN of $\mu = 0$, SNR=20dB and N=6000.

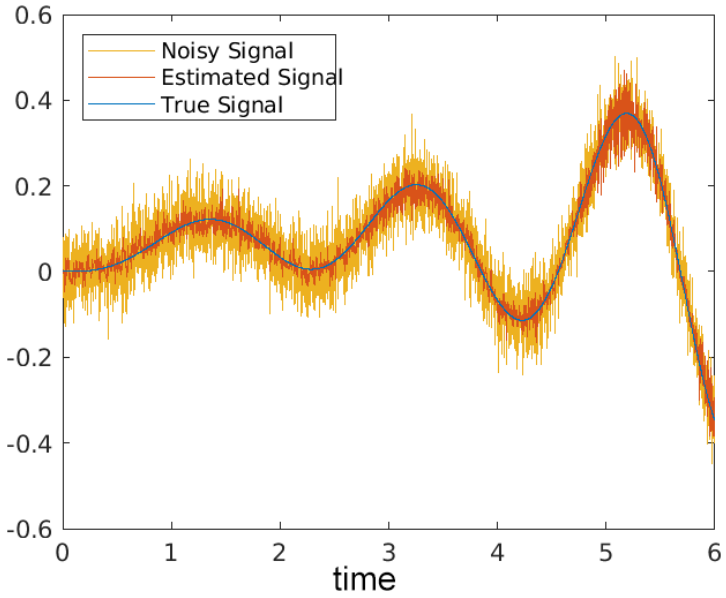

Figure 6.8: Noisy, true and estimated 4th order linear system outputs by NLKDS with AWGN of $\mu = 0$, SNR=10dB and N=6000.

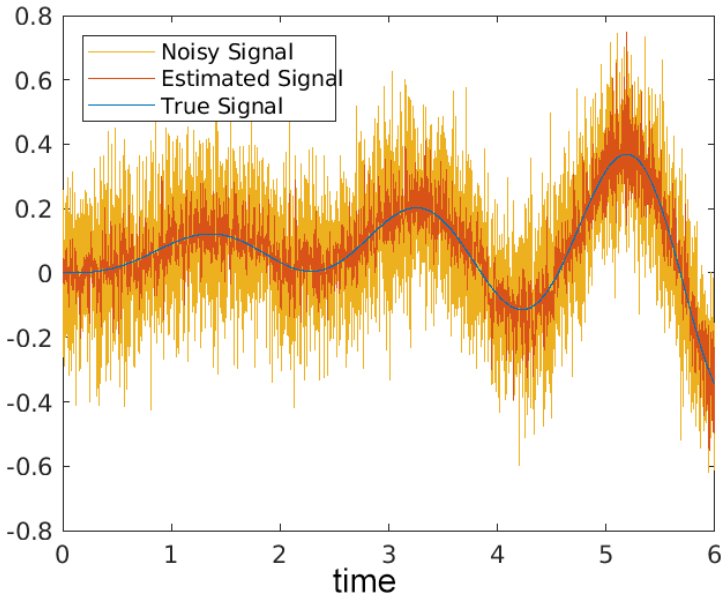

Figure 6.9: Noisy, true and estimated 4th order linear system outputs by NLKDS with AWGN of $\mu = 0$, SNR=0dB and N=6000.

		Linear KDS	KKF-CEO	NLKDS
20dB	MSE	0.0029	0.0150	0.0041
	SNR	30.2343dB	23.0537dB	27.2926dB
	Runtime	76.6128s	36.9109s	10.1574s
10dB	MSE	0.0334	0.1055	0.0443
	SNR	19.5740dB	14.5829dB	16.9348dB
	Runtime	122.5762s	36.8121s	10.3588s
0dB	MSE	0.0981	0.6929	0.3701
	SNR	14.8994dB	6.4088dB	7.7712dB
	Runtime	790.3135s	36.3670s	10.5786s

Table 6.1: The performances of compared algorithms on linear system.

6.2 Performances on Nonlinear Systems

Because the linear KDS algorithm can only estimate the true outputs $\{x(t_i)\}$ of linear dynamical systems, we only compare our algorithm's performances with KKF-CEO. We estimate the true outputs of Van Der Pol equation, Sedoglavic equation and IKEDA chaotic dynamical system under different noise levels in our experiments.

6.2.1 Van Der Pol Equation

Firstly, we perform the estimation on Van Der Pol equation, the governing equation of the system is:

$$\begin{aligned} x^{(2)}(t) &= \mu(1-x^2)x^{(1)}(t) + x(t) \\ y(t) &= x(t) + \nu(t) \end{aligned} \tag{6.4}$$

The value of the parameter μ is 0.5, the initial condition $\left[\begin{smallmatrix} x^{(1)}(0) & x(0) \end{smallmatrix}\right]$ is $\left[\begin{smallmatrix} 1 & 1 \end{smallmatrix}\right]$. The noisy measurements $\{y(t_i)\}$ are generated by adding white Gaussian noise with mean $\mu = 0$ and $SNR = \{-10, -3, 0, 3, 10, 20\}dB$ to the true outputs $\{x(t_i)\}$. The time interval is $[0, 10]$, the number of sample points N is 2000. The size of measurements is reduced to 400

in KKF-CEO, while our novel algorithm do not need to reduce the size of measurements. The variance of the Gaussian distribution σ is set to the median of pairwise distance of noisy measurements in both algorithms. The values of q are tuned to 1 and the values of r are tuned to $\{3, 2, 1.5, 1.25, 1, 0.5\}$ corresponding to $SNR = \{20, 10, 3, 0, -3, -10\}dB$, ζ and λ are set to 10^{-3} and 10^{-4} in KKF-CEO.

Figure 6.10-6.15 are the plots of the noisy measurements, the true outputs and the outputs estimated by KKF-CEO, and Figure 6.16-6.21 are the plots of noisy measurements, the true outputs and the outputs estimated by our algorithm. Table 6.2 shows the performances of compared algorithms under different noise levels.

From the results, we can see that for the Van Der Pol equation, both the accuracy and runtime of our novel algorithm outperforms KKF-CEO. For accuracy, our novel algorithm gains around 8dB of SNR from the noisy measurements while KKF-CEO gains around 4.5dB of SNR from the noisy measurements. Our algorithm gains around 3.5dB more SNR than KKF-CEO. For time complexity, the runtime of our algorithm is around 20 times faster than KKF-CEO, even if the size of data points we used during the estimation is five times as big as the size of the data used by KKF-CEO. Our process of reducing time complexity has a massive effect here.

	MSE		SNR		Runtime	
	KKF-CEO	NLKDS	KKF-CEO	NLKDS	KKF-CEO	NLKDS
20dB	0.0072	0.0024	23.6558dB	28.4090dB	18.5640s	1.2762s
10dB	0.0590	0.0228	14.5067dB	18.6203dB	18.5069s	0.8549s
3dB	0.2945	0.1377	7.5275dB	10.8178dB	18.9370s	0.9470s
0dB	0.6155	0.2720	4.3262dB	7.8616dB	19.4463s	1.0821s
-3dB	1.4129	0.4894	1.6384dB	5.3707dB	21.9863s	0.9696s
-10dB	8.9494	2.1147	-7.2994dB	-1.0453dB	19.0911s	1.0818s

Table 6.2: The performances of compared algorithms under different noise levels on Van Der Pol equation.

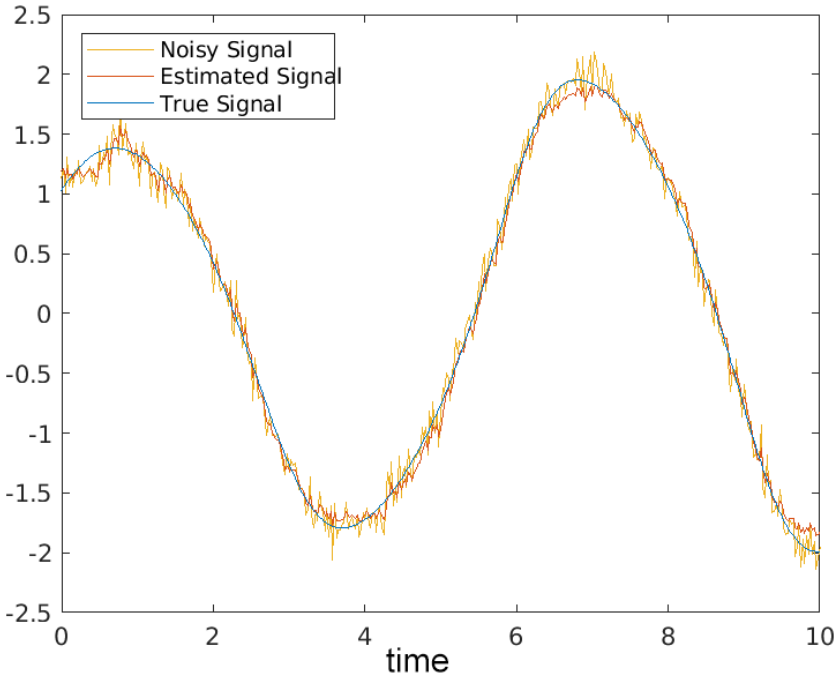

Figure 6.10: Noisy, true and estimated Van Der Pol equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=20dB.

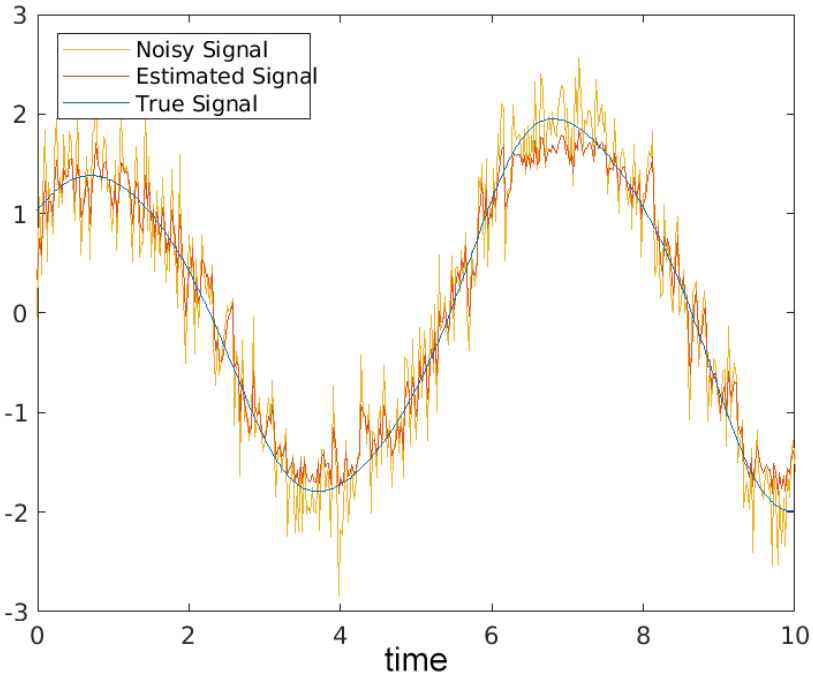

Figure 6.11: Noisy, true and estimated Van Der Pol equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=10dB.

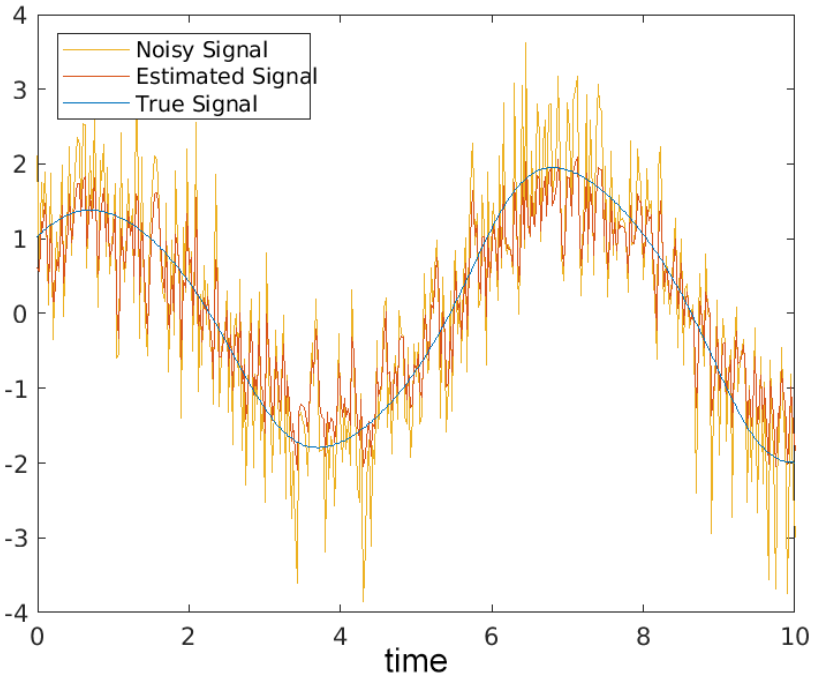

Figure 6.12: Noisy, true and estimated Van Der Pol equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=3dB.

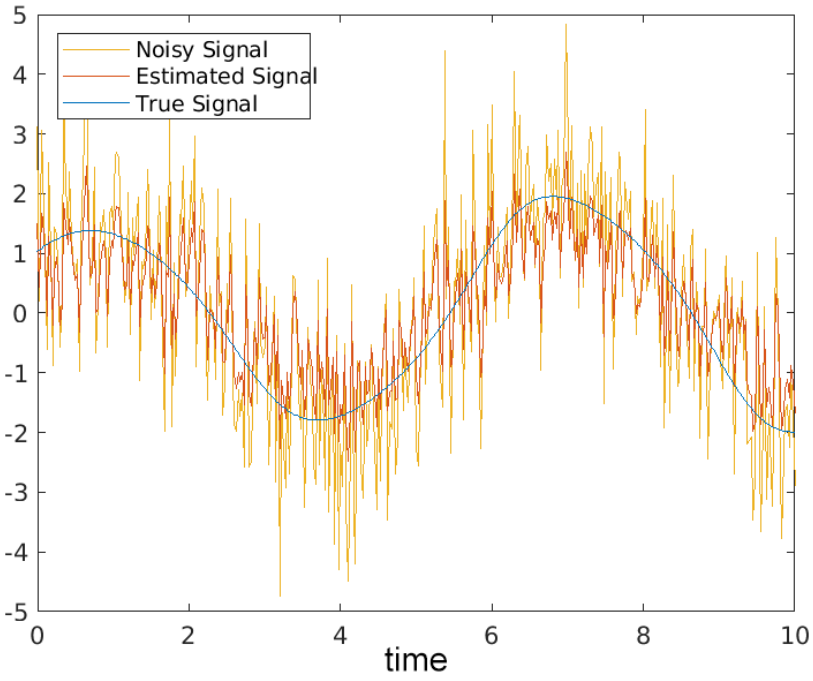

Figure 6.13: Noisy, true and estimated Van Der Pol equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=0dB.

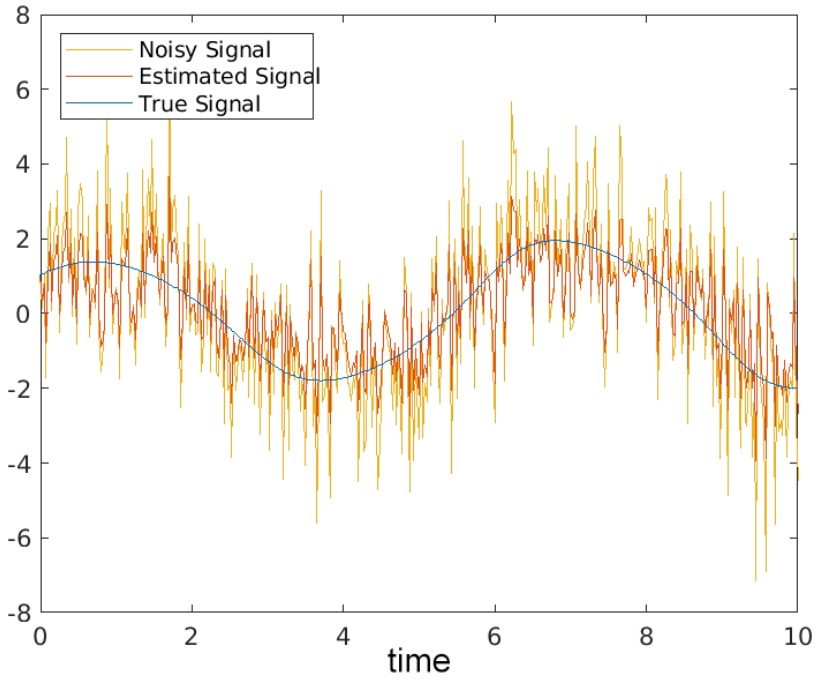

Figure 6.14: Noisy, true and estimated Van Der Pol equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=-3dB.

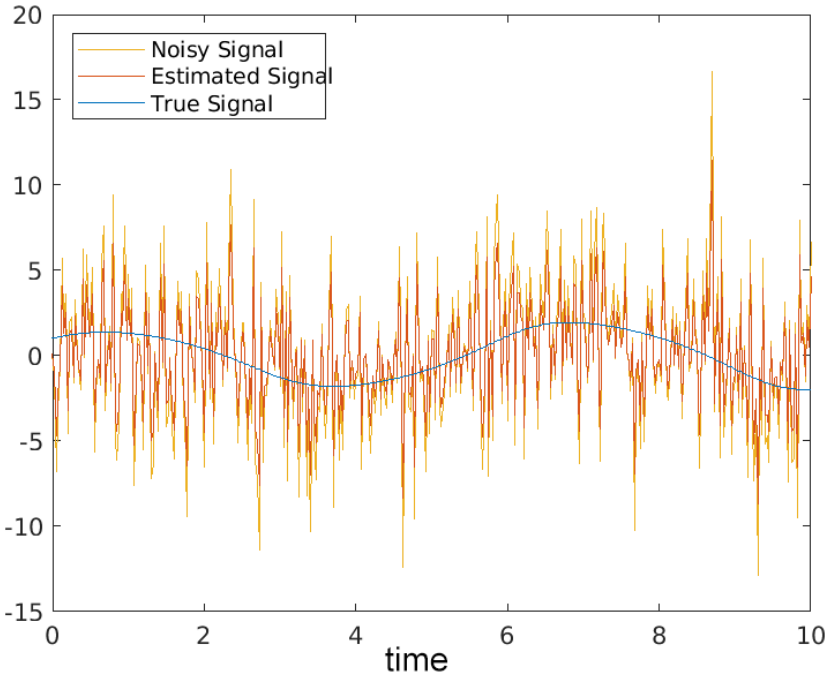

Figure 6.15: Noisy, true and estimated Van Der Pol equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=-10dB.

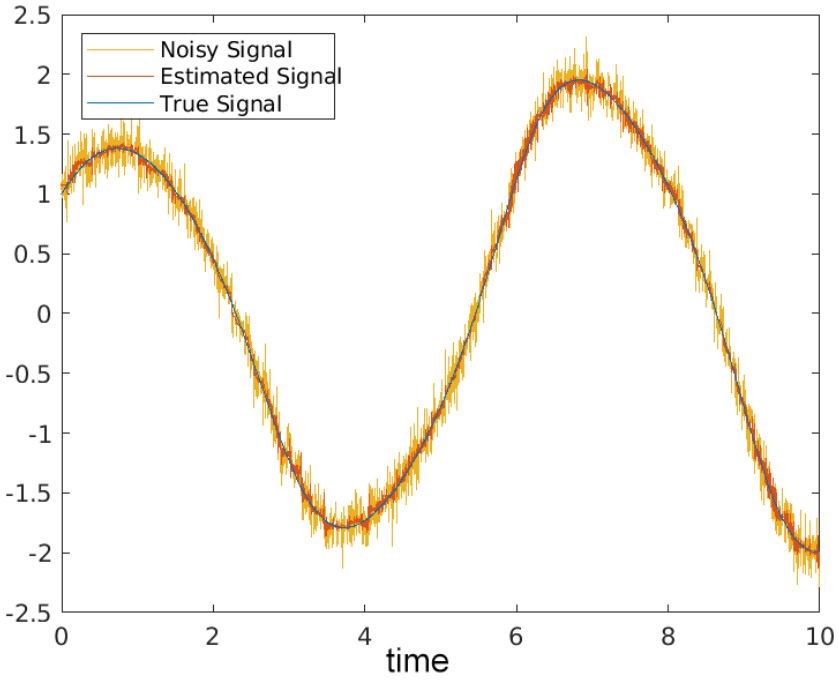

Figure 6.16: Noisy, true and estimated Van Der Pol equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=20dB.

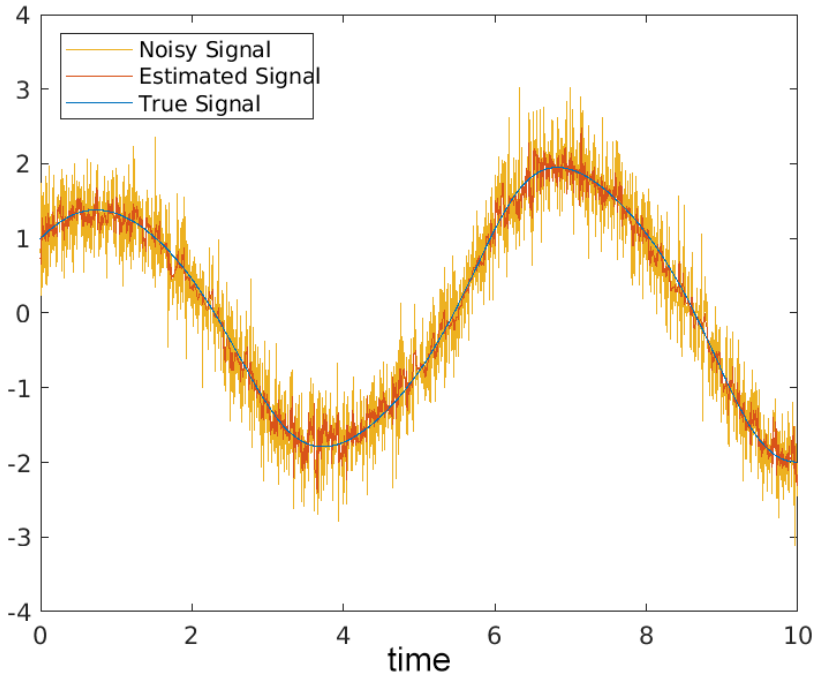

Figure 6.17: Noisy, true and estimated Van Der Pol equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=10dB.

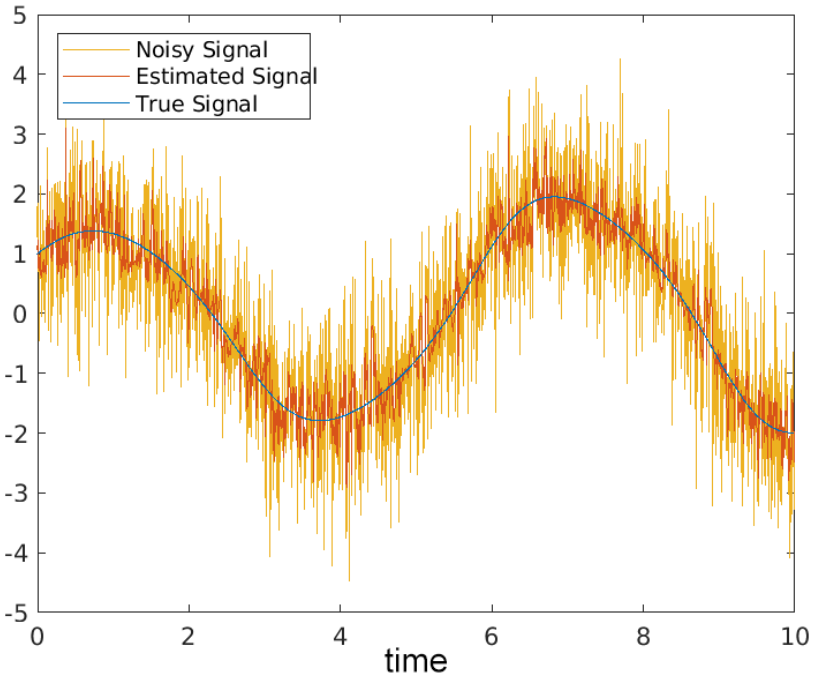

Figure 6.18: Noisy, true and estimated Van Der Pol equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=3dB.

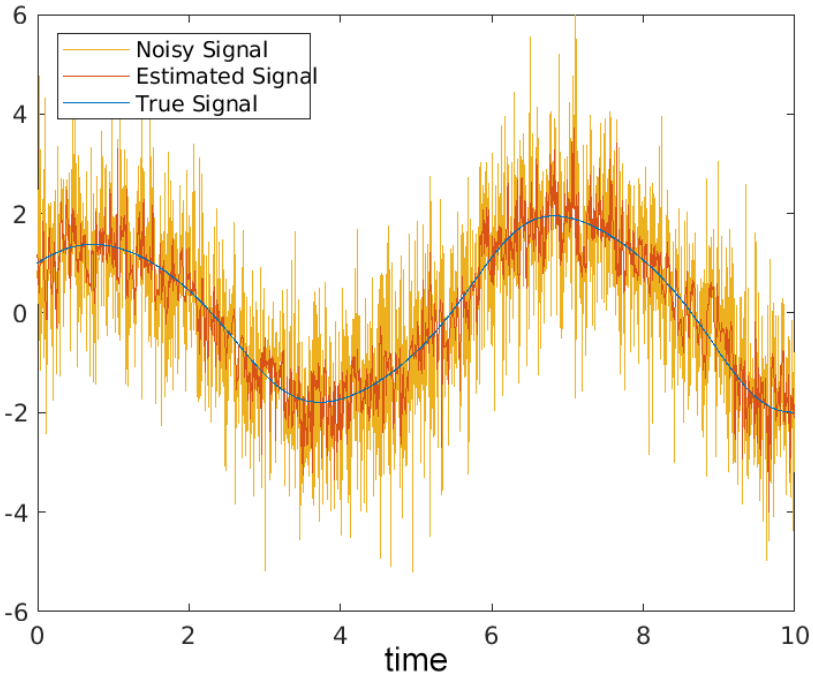

Figure 6.19: Noisy, true and estimated Van Der Pol equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=0dB.

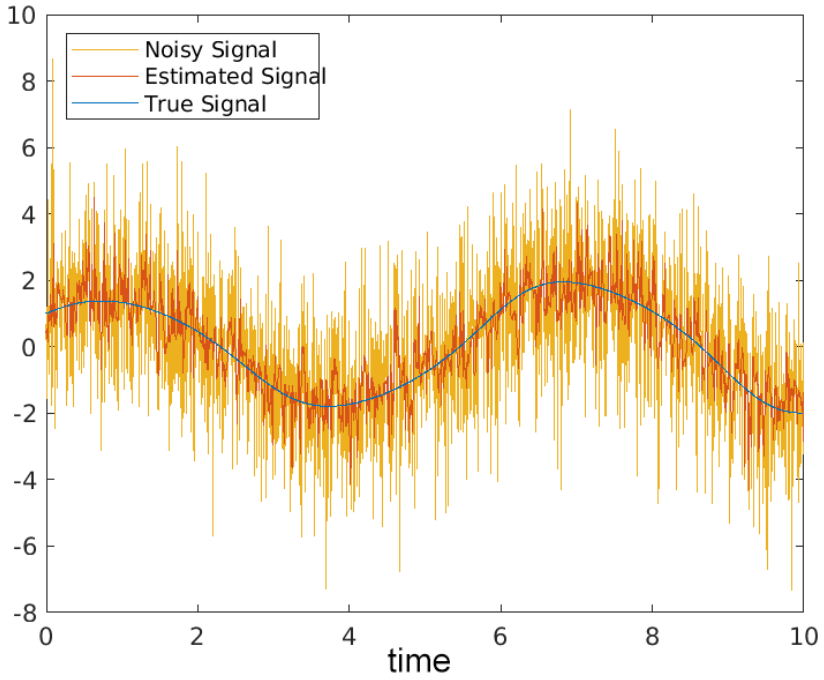

Figure 6.20: Noisy, true and estimated Van Der Pol equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=-3dB.

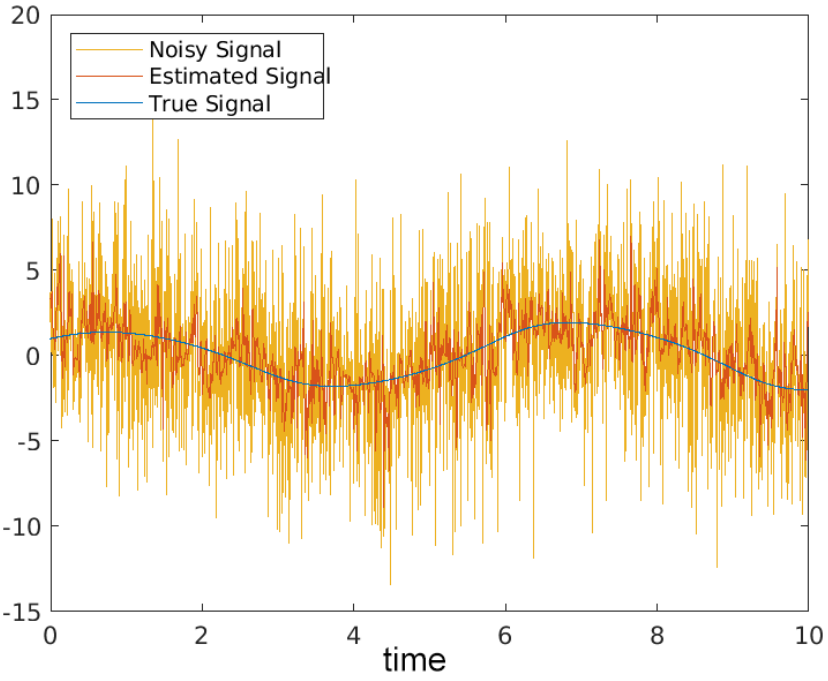

Figure 6.21: Noisy, true and estimated Van Der Pol equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=-10dB.

6.2.2 Sedoglavic Equation

In this section, we perform the estimation on Sedoglavic equation, the governing equation of the system is:

$$\begin{aligned} \dot{x}_1(t) &= \frac{x_2(t)}{x_1(t)} \\ \dot{x}_2(t) &= \frac{x_3(t)}{x_2(t)} \\ \dot{x}_3(t) &= \theta x_1(t) \end{aligned} \tag{6.5}$$

The value of the parameter θ is 0.5, the initial condition $[\, x_3(0)\ x_2(0)\ x_1(0) \,]$ is $[\, 1\ 1\ 1 \,]$. The noisy measurements $\{y(t_i)\}$ are generated by adding white Gaussian noise with mean $\mu = 0$ and $SNR = \{-10, -3, 0, 3, 10, 20\} dB$ to the true outputs $\{x(t_i)\} = \{x_1(t_i)\}$. The time interval is $[0, 10]$, the number of sample points N is 2000. The data size reducing process and the values of the parameters are same as the experiment on Van Der Pol equation. Figure 6.22-6.33 are the plots of noisy measurements, the true outputs and the outputs estimated by compared algorithms. Table 6.3 shows the performances of compared algorithms under different noise levels. For Sedoglavic equation, the accuracy of our algorithm also outperforms KKF-CEO, especially under the noisy measurements with low SNR. Our algorithm gains around 5dB more SNR under -10dB noisy measurement. And our algorithm also runs 20 times faster than KKF-CEO using four times more data.

	MSE		SNR		Runtime	
	KKF-CEO	NLKDS	KKF-CEO	NLKDS	KKF-CEO	NLKDS
20dB	0.1843	0.1486	24.3391dB	25.1526dB	18.4695s	1.0026s
10dB	2.1667	1.4520	13.6365dB	15.2538dB	20.3651s	1.0200s
3dB	8.2623	4.9173	7.8236dB	9.9121dB	17.0213s	0.8980s
0dB	13.4219	8.7062	5.7165dB	7.4750dB	15.5402s	0.9383s
-3dB	35.2860	15.0315	1.5186dB	5.1033dB	17.9967s	0.9583s
-10dB	239.0310	75.0232	-6.7899dB	-1.8787dB	18.6236s	0.9263s

Table 6.3: The performances of compared algorithms under different noise levels on Sedoglavic equation.

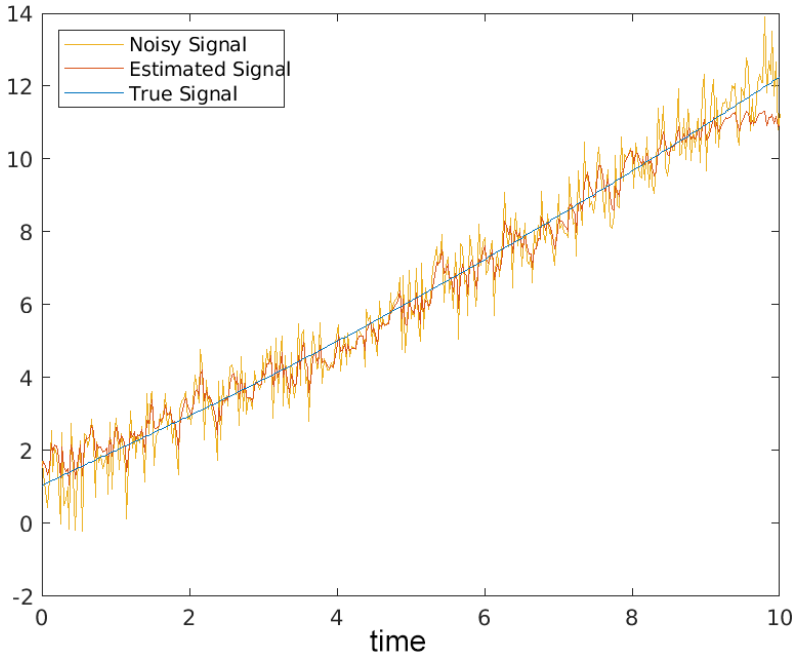

Figure 6.22: Noisy, true and estimated Sedoglavic equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=20dB.

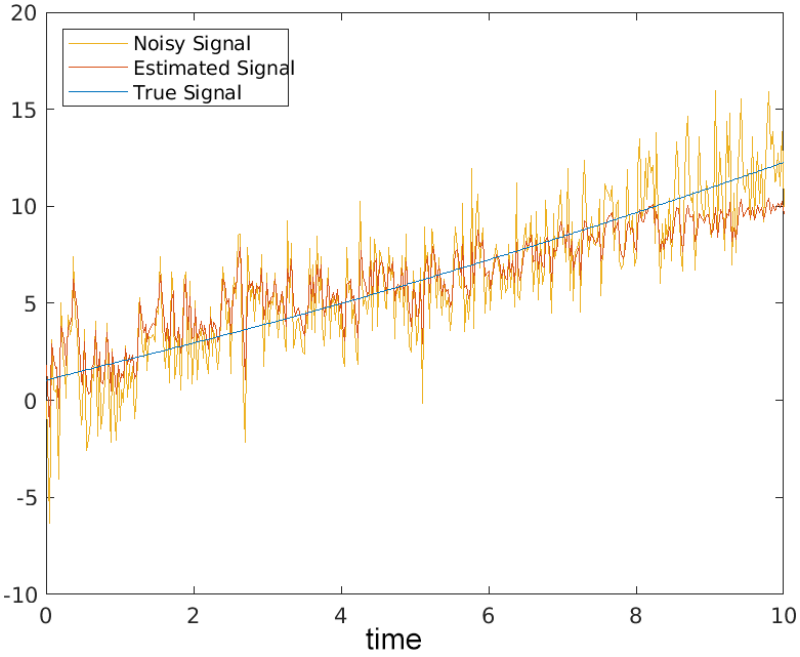

Figure 6.23: Noisy, true and estimated Sedoglavic equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=10dB.

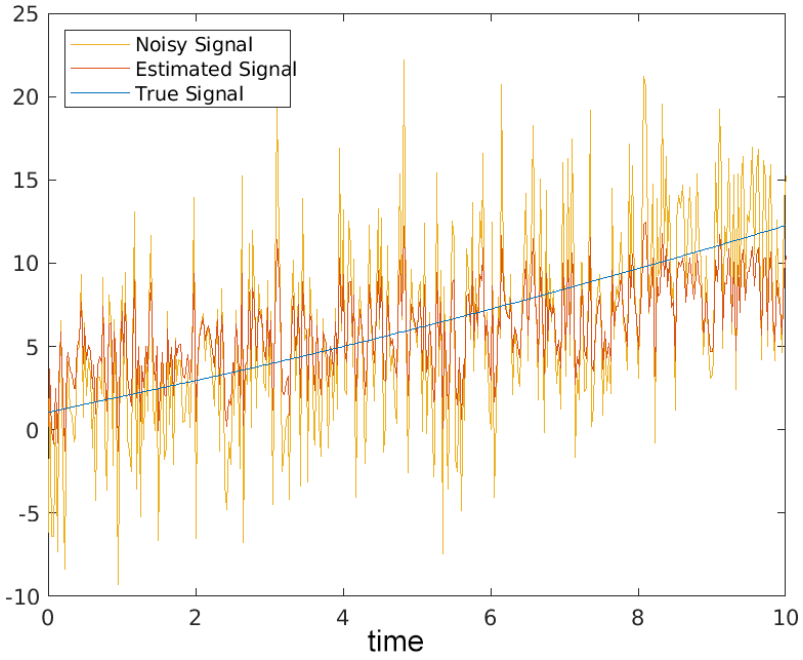

Figure 6.24: Noisy, true and estimated Sedoglavic equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=3dB.

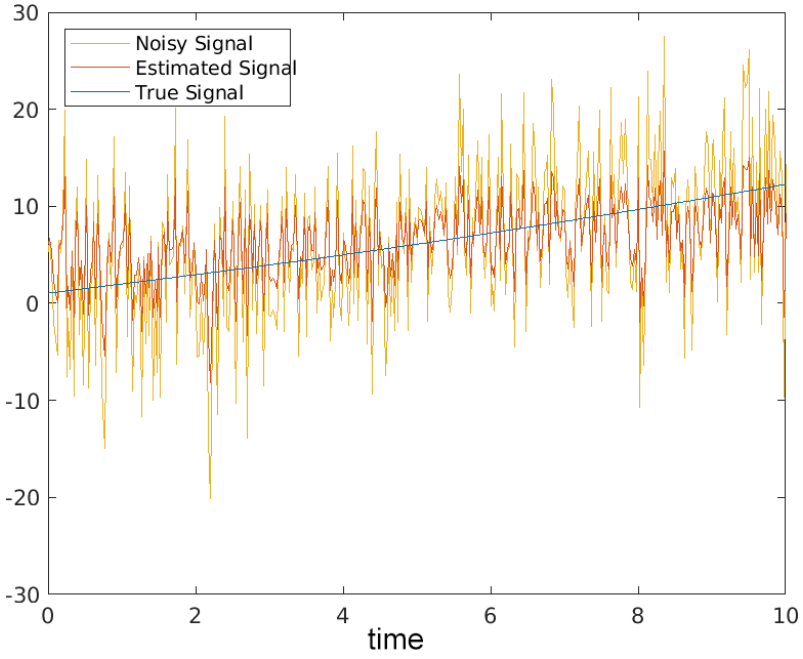

Figure 6.25: Noisy, true and estimated Sedoglavic equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=0dB.

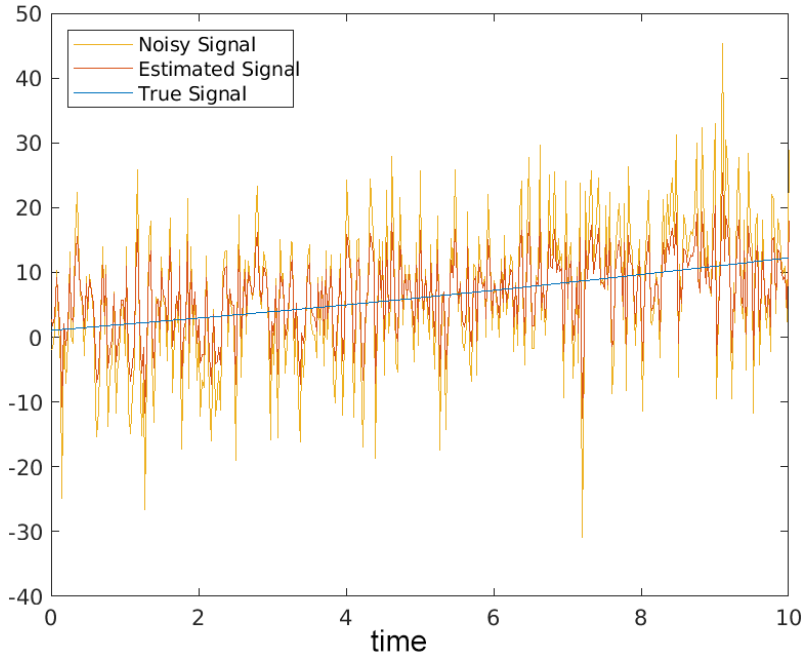

Figure 6.26: Noisy, true and estimated Sedoglavic equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=-3dB.

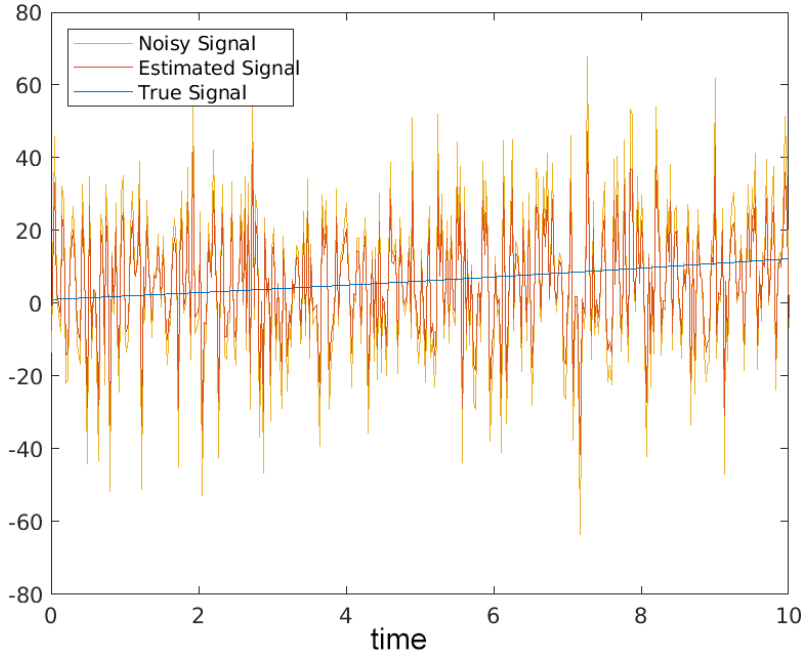

Figure 6.27: Noisy, true and estimated Sedoglavic equation outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=-10dB.

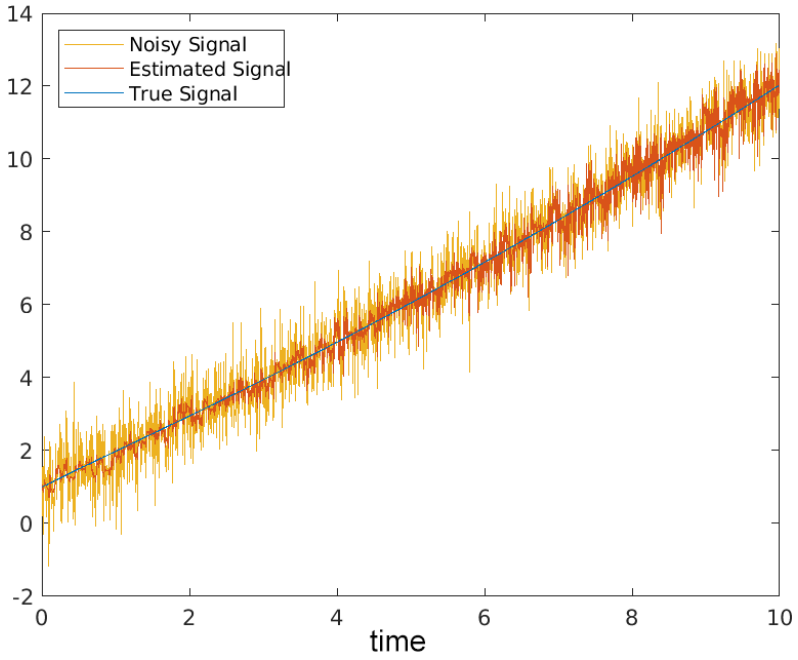

Figure 6.28: Noisy, true and estimated Sedoglavic equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=20dB.

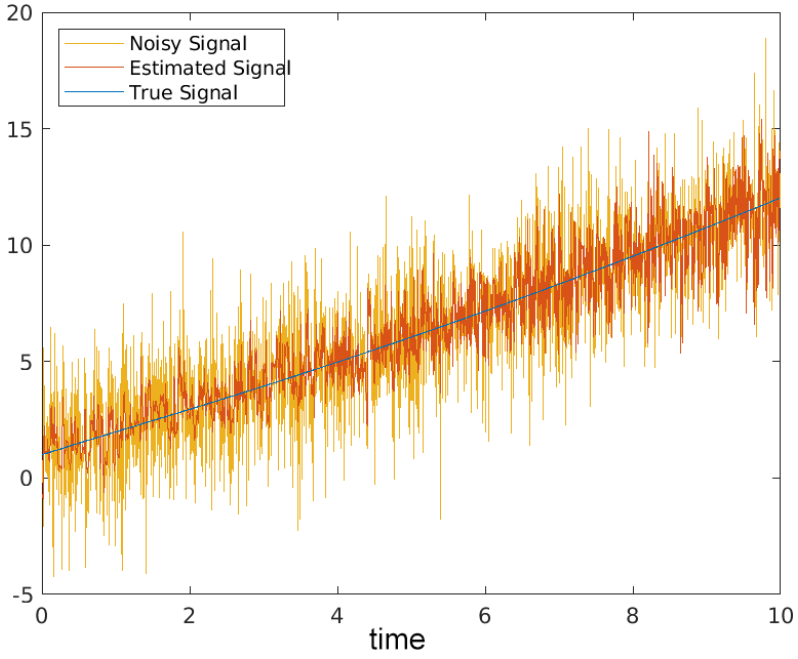

Figure 6.29: Noisy, true and estimated Sedoglavic equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=10dB.

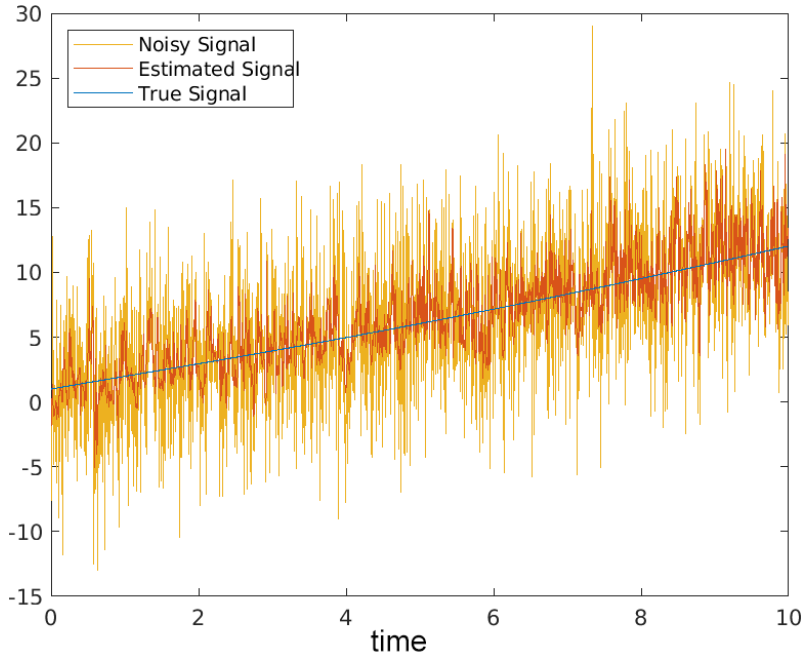

Figure 6.30: Noisy, true and estimated Sedoglavic equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=3dB.

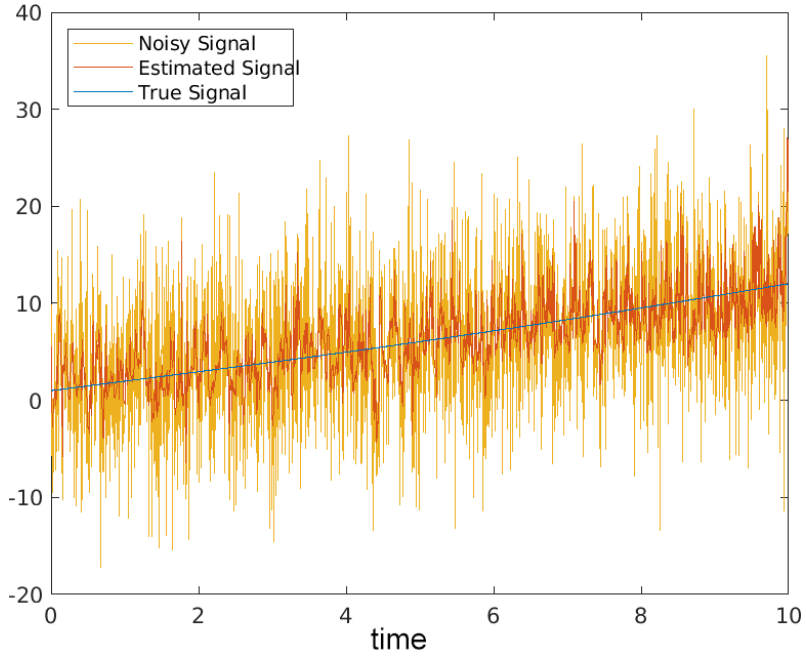

Figure 6.31: Noisy, true and estimated Sedoglavic equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=0dB.

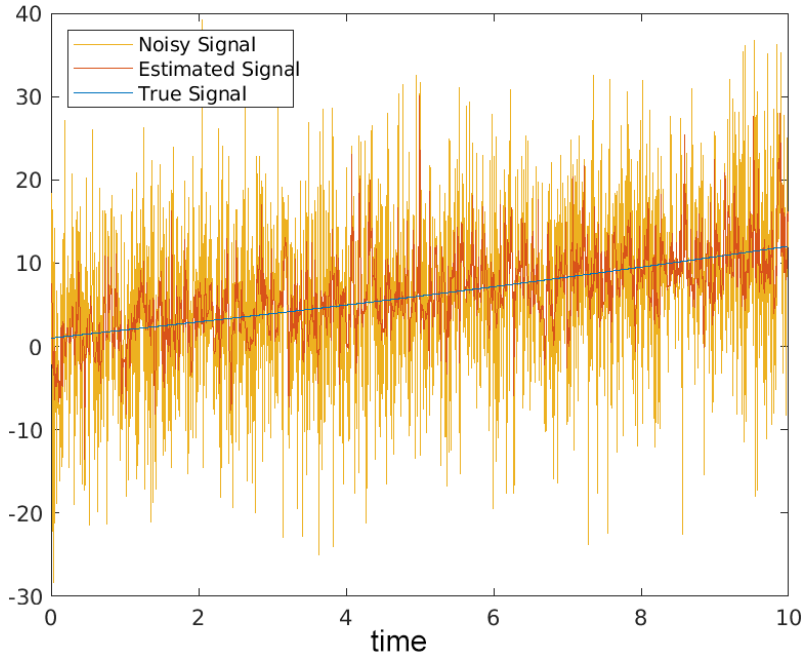

Figure 6.32: Noisy, true and estimated Sedoglavic equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=-3dB.

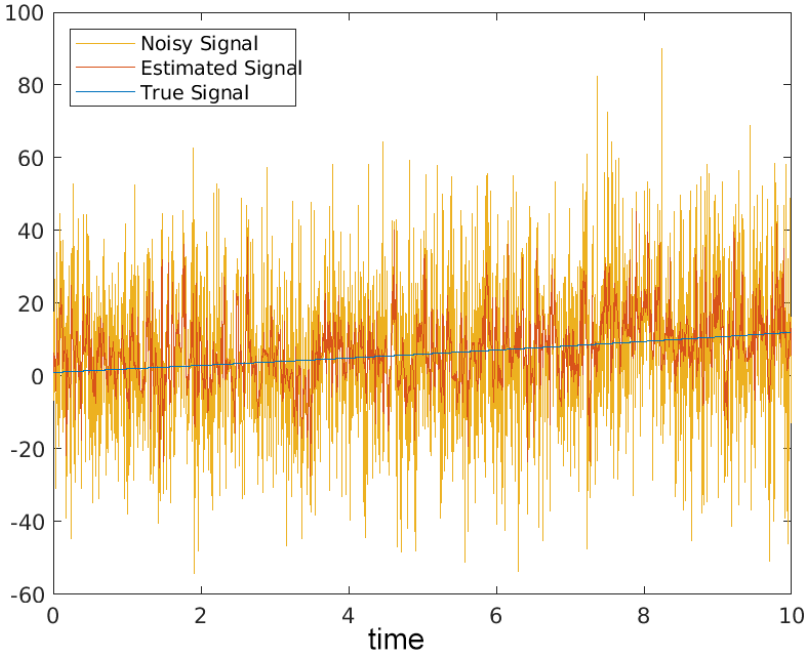

Figure 6.33: Noisy, true and estimated Sedoglavic equation outputs by nonlinear KDS with AWGN of $\mu = 0$, SNR=-10dB.

6.2.3 IKEDA Chaotic Dynamical System [12]

Here, we perform the estimation on IKEDA chaotic dynamical system, which is more complicated, the governing equation of the system is:

$$\begin{cases} w(t) = c_1 - \frac{c_3}{1+x_1(t)^2+x_2(t)^2} \\ x_1(t+1) = c_4 + c_2(x_1(t)\cos(w(t)) - x_2(t)\sin(w(t))) \\ x_2(t+1) = c_2(x_1(t)\sin(w(t)) + x_2(t)\cos(w(t))) \end{cases} \tag{6.6}$$

The parameters c_1, c_2, c_3 and c_4 are set to $c_1 = 0.4$, $c_2 = 0.84$, $c_3 = 6.0$, $c_4 = 1.0$, the initial condition $\begin{bmatrix} x_1(0) & x_2(0) \end{bmatrix}$ is $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$. The noisy measurements $\{y(t_i)\}$ are generated by adding white Gaussian noise with mean $\mu = 0$ and $SNR = \{-3, 0, 3\}dB$ to the true outputs $\{x(t_i)\} = \{x_1(t_i)\}$. The time interval is $[0, 200]$, the number of sample points N is 200. The data size reducing process is not needed in this section.

Figure 6.34-6.36 are the plots of noisy measurements, the true outputs and the outputs estimated by KKF-CEO, and Figure 6.37-6.39 are the plots of noisy measurements, the true outputs and the outputs estimated by our novel algorithm. Table 6.4 shows the performances of compared algorithms under different noise levels.

From the results, we can see that for systems as complicated as IKEDA chaotic dynamical system, the accuracy of our algorithm cannot outperform KKF-CEO. Our algorithm gains around 3dB less SNR than KKF-CEO. However, the runtime of our algorithm is 100 times shorter than KKF-CEO using the same size of data points as KKF-CEO.

	MSE		SNR		Runtime	
	KKF-CEO	NLKDS	KKF-CEO	NLKDS	KKF-CEO	NLKDS
3dB	0.1001	0.2350	7.8589dB	4.1901dB	3.8878s	0.0331s
0dB	0.1685	0.2467	5.5962dB	3.9784dB	4.1499s	0.0378s
-3dB	0.1946	0.3539	4.9707dB	2.4107dB	6.6530s	0.0346s

Table 6.4: The performances of compared algorithms under different noise levels on IKEDA chaotic dynamical system.

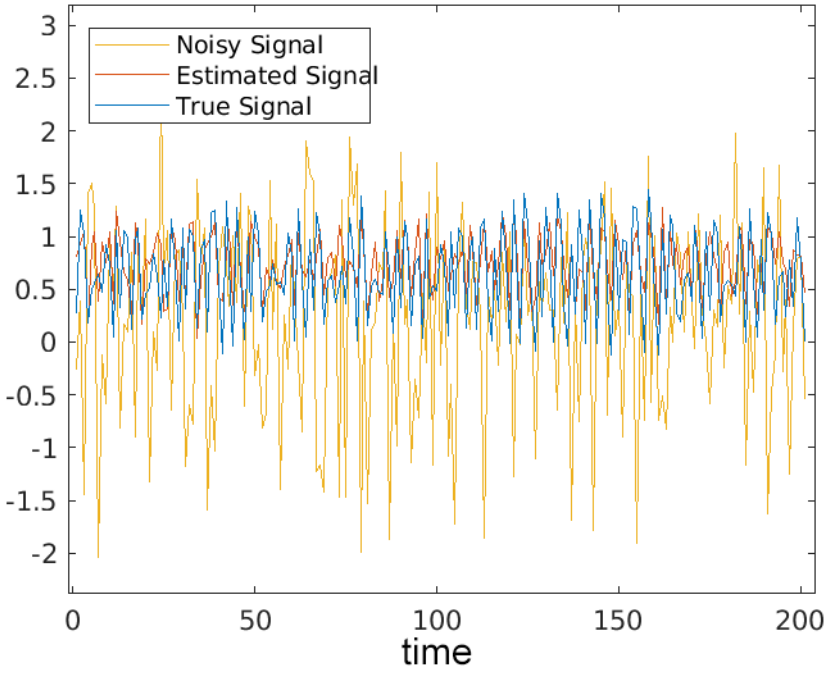

Figure 6.34: Noisy, true and estimated IKEDA chaotic dynamical system outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=3dB.

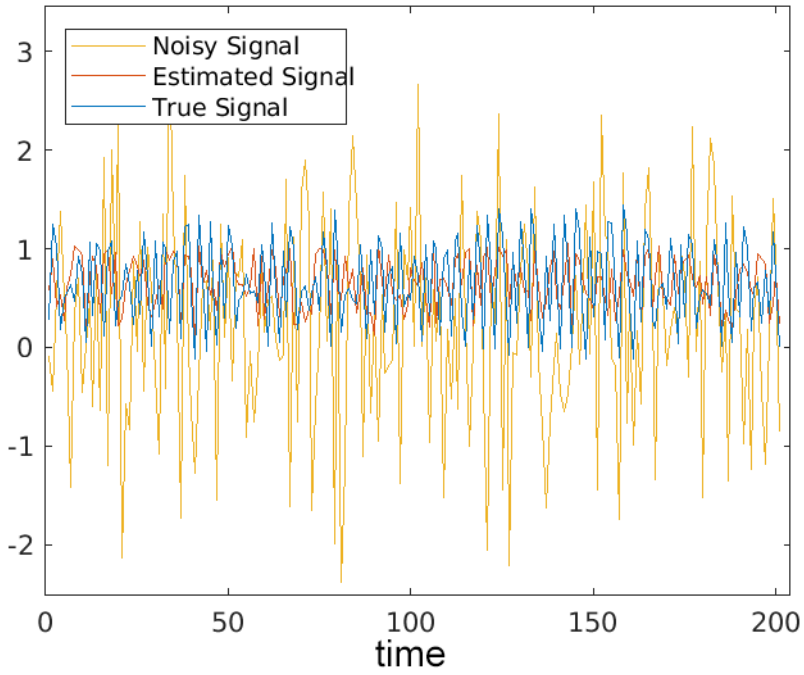

Figure 6.35: Noisy, true and estimated IKEDA chaotic dynamical system outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=0dB.

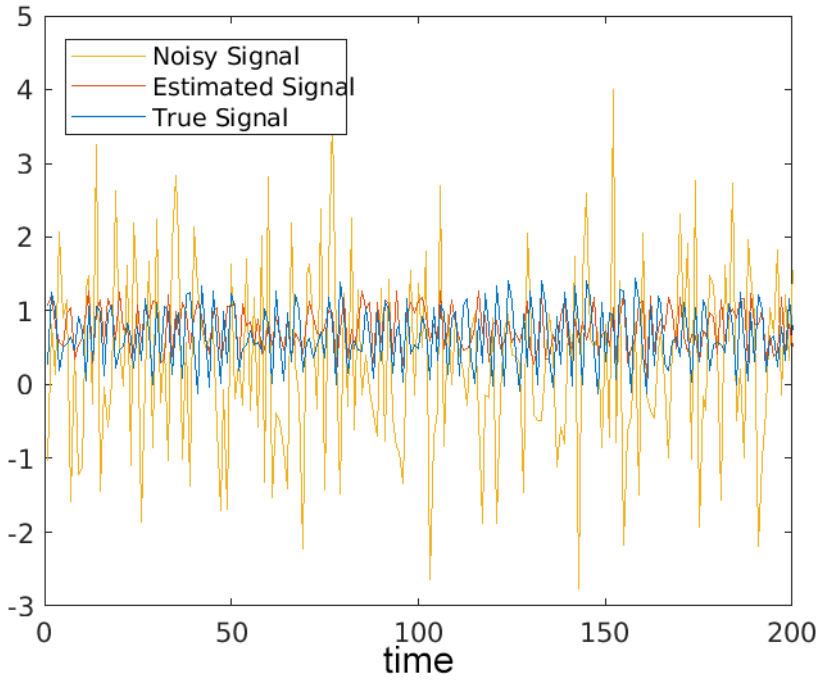

Figure 6.36: Noisy, true and estimated IKEDA chaotic dynamical system outputs by KKF-CEO with AWGN of $\mu = 0$, SNR=-3dB.

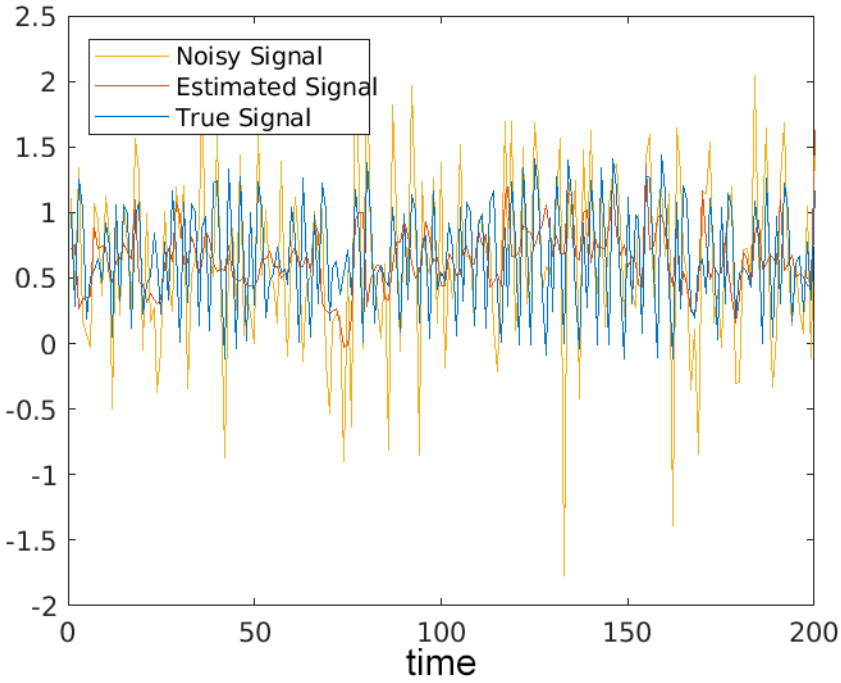

Figure 6.37: Noisy, true and estimated IKEDA chaotic dynamical system outputs by non-linear KDS with AWGN of $\mu = 0$, SNR=3dB.

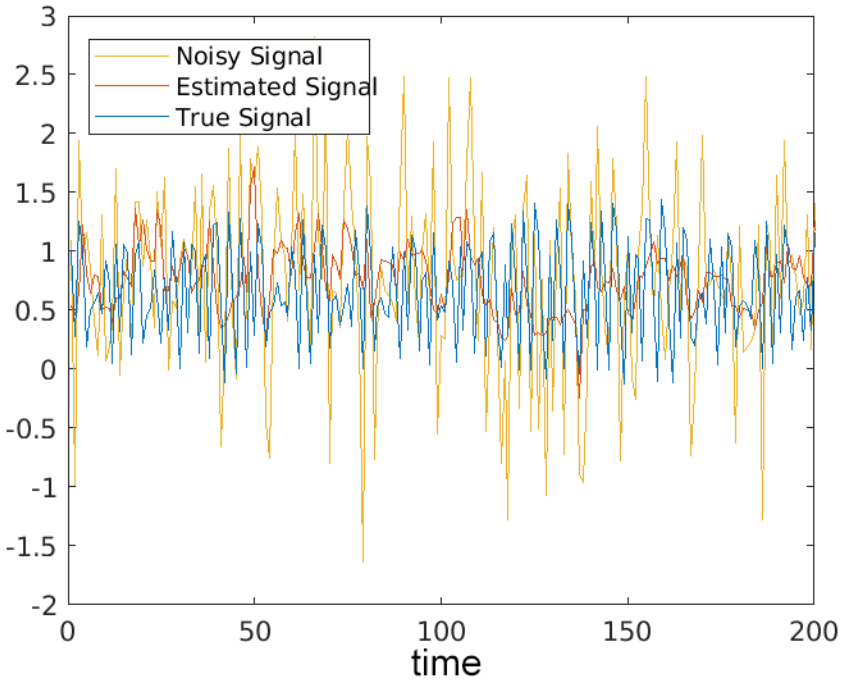

Figure 6.38: Noisy, true and estimated IKEDA chaotic dynamical system outputs by non-linear KDS with AWGN of $\mu = 0$, SNR=0dB.

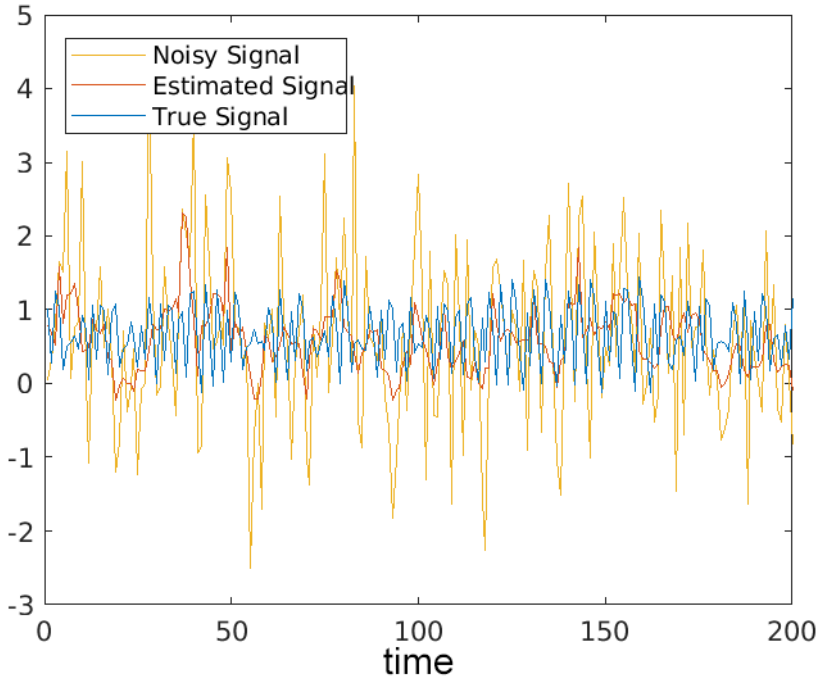

Figure 6.39: Noisy, true and estimated IKEDA chaotic dynamical system outputs by non-linear KDS with AWGN of $\mu = 0$, SNR=-3dB.

Chapter 7

Discussion and Conclusion

In this thesis, a novel algorithm which is adapting KDS introduced in Chapter 3 for nonlinear dynamical systems inspired by KKF-CEO introduced in Chapter 4 is proposed by us.

In this thesis, the problem we focused on is to estimate the output of a nonlinear dynamical system using noisy measurements without knowing the underlying system. However, most of the existing algorithms for nonlinear dynamical systems such as the EKF, UKF, DMSCE and KBR require the knowledge of the underlying system or a clean training set. While in real life, both the knowledge of the underlying system and the clean training set are unavailable. KKF-CEO proposed by Zhu et al. [35] which is based on classical Kalman filter can deal with the proposed estimation problem, while KDS proposed by John et al. [15] outperforms the Kalman filter. Therefore, our novel algorithm adapts KDS for nonlinear systems using a similar method to Zhu et al.

Firstly our novel algorithm is applied to estimate the outputs of fourth order linear system. The performances under different noise levels are compared with linear KDS and KKF-CEO. From the numerical results in Table 6.1, we can find that the linear KDS has the best accuracy, while our novel algorithm has the second-best accuracy. However, the runtime of our novel algorithm is the shortest. In addition, our novel algorithm can be applied to both linear and nonlinear systems, while linear KDS can only be applied to

linear systems.

Our novel algorithm is then applied to estimate the outputs of nonlinear systems, three nonlinear systems: Van Der Pol equation, Sedoglavic equation and IKEDA chaotic dynamical system are implemented. From the numerical results in Table 6.2 and Table 6.3, we can find that both accuracy and runtime performances of our novel algorithm outperform KKF-CEO on Van Der Pol equation and Sedoglavic equation. However, from Table 6.4, we find that the accuracy performance of our novel algorithm on IKEDA chaotic dynamical system is worse than KKF-CEO.

The reason our new algorithm performs worse than KKF-CEO on IKEDA chaotic dynamical system is IKEDA chaotic dynamical system is more complicated than Van Der Pol equation and Sedoglavic equation. A higher order approximation use instead of (5.3) may provide a better result.

The reason that the time complexity of our new algorithm outperforms the other two algorithms is the simplification of summation implemented by us in (5.36), which reduces the time complexity of our algorithm form $O(n^3)$ to $O(n^2)$. In contrast, the time complexity of KKF-CEO is $O(n^3)$. When the size of the data set is large, a downsampling process needs to be implemented in KKF-CEO, while our novel algorithm does not require the downsampling process. In addition, the total time complexity of the downsampling and estimation process in KKF-CEO is $O(k^3)+O(n^2)+O(nk)$, the time complexity of our novel algorithm still outperforms KKF-CEO with downsampling.

Another advantage of our novel algorithm compared to KKF-CEO is no parameter is needed to be tuned in our novel algorithm, while parameters q and r need to be tuned based on the prior knowledge of the noise to achieve good performances in KKF-CEO.

In the future, a similar algorithm based on a higher order approximation instead of (5.3) can be developed to get better results. In addition, We can adapt our algorithm for multiple input multiple output nonlinear dynamical systems.

Appendix A

Kalman Filter

Algorithm 4: Kalman Filter [19]

Initialization: For $i = 0$, set
$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]$
$\mathbf{P}_0 = E[(\mathbf{x}_0 - E[\mathbf{x}_0)(\mathbf{x}_0 - E[\mathbf{x}_0)^T]$

Computation: For $i = 1, ...,$ compute
State estimate propagation:
$\hat{\mathbf{x}}_i^- = \mathbf{F}_{i-1}\hat{\mathbf{x}}_{i-1}$
Measurement prediction:
$\hat{\mathbf{y}}_i = \mathbf{H}_i\hat{\mathbf{x}}_i^-$
Error covariance propagation:
$\mathbf{P}_i^- = \mathbf{F}_{i-1}\mathbf{P}_{i-1}\mathbf{F}_{i-1}^T + \mathbf{Q}_{i-1}$
Kalman gain matrix:
$\mathbf{G}_i = \mathbf{P}_i^-\mathbf{H}_i^T[\mathbf{H}_i\mathbf{P}_i^-\mathbf{H}_i^T + \mathbf{R}_i]^{-1}$
State estimate update:
$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_i^- + \mathbf{G}_i(\mathbf{y}_i - \mathbf{H}_i\hat{\mathbf{x}}_i^-)$
Error covariance update:
$\mathbf{P}_i = (\mathbf{I} - \mathbf{G}_i\mathbf{H}_i)\mathbf{P}_i^-$

Appendix B

Proofs of Theorem 4.2.1 and Theorem 4.2.2 [35]

Because Theorem 4.2.1 and 4.2.2 are closely related, they are proved together here.

Proof: Given $\mathbf{F}_i = \Phi(\mathbf{K} + \zeta n \mathbf{I}_n)^{-1}\Upsilon^T$, $\mathbf{H}_i = \mathbf{I}$, $\mathbf{Q}_i = q\mathbf{I}$ and $\mathbf{R}_i = r\mathbf{I}$. Start with $\mu_0 = \varphi(\mathbf{y}_0)$, $\mathbf{P}_0 = \lambda\mathbf{I}$, substitute them into (4.30) to (4.34), we have the first iteration:

μ_1^- is predicted as

$$\mu_1^- = \mathbf{F}_0\mu_0 = \Phi(\mathbf{K} + \zeta n \mathbf{I}_n)^{-1}\Upsilon^T\mu_0 = \Phi\mathbf{a}_1 \tag{B.1}$$

Then, $\mathbf{P}_1^-$ can be expressed as

$$\begin{aligned}\mathbf{P}_1^- &= \mathbf{F}_0\mathbf{P}_0\mathbf{F}_0^T + q\mathbf{I} \\ &= \lambda\Phi(\mathbf{K} + \zeta n \mathbf{I}_n)^{-1}\Upsilon^T\Upsilon[(\mathbf{K} + \zeta n \mathbf{I}_n)^{-1}]^T\Phi^T + q\mathbf{I} \\ &= \Phi\tilde{\mathbf{P}}_1^-\Phi^T + q\mathbf{I}\end{aligned} \tag{B.2}$$

Kalman gain $\mathbf{G}_1$ is

$$\begin{aligned}\mathbf{G}_1 =& \mathbf{P}_1^-\mathbf{H}_1^T[\mathbf{H}_1\mathbf{P}_1^-\mathbf{H}_1^T + \mathbf{R}_1]^{-1} \\ =& \mathbf{P}_1^-[\mathbf{P}_1^- + r\mathbf{I}]^{-1}\end{aligned}$$

$$
\begin{aligned}
&=(\Phi\tilde{\mathbf{P}}_1^-\Phi^T+q\mathbf{I})[\Phi\tilde{\mathbf{P}}_1^-\Phi^T+(q+r)\mathbf{I}]^{-1}\\
&=\left[\frac{q}{q+r}[\Phi\tilde{\mathbf{P}}_1^-\Phi^T+(q+r)\mathbf{I}]+\frac{r}{q+r}\Phi\tilde{\mathbf{P}}_1^-\Phi^T\right]\\
&\quad\times[\Phi\tilde{\mathbf{P}}_1^-\Phi^T+(q+r)\mathbf{I}]^{-1}\\
&=\frac{r}{q+r}\Phi\tilde{\mathbf{P}}_1^-\Phi^T[\Phi\tilde{\mathbf{P}}_1^-\Phi^T+(q+r)\mathbf{I}]^{-1}+\frac{q}{q+r}\mathbf{I}
\end{aligned}
\tag{B.3}
$$

Then, from matrix inversion lemma [9],

$$
\begin{aligned}
\mathbf{G}_1=&\frac{r}{q+r}\Phi[(q+r)\mathbf{I}_m+\tilde{\mathbf{P}}_1^-\Phi^T\Phi]^{-1}\tilde{\mathbf{P}}_1^-\Phi^T+\frac{q}{q+r}\mathbf{I}\\
=&\frac{r}{q+r}\Phi\tilde{\mathbf{G}}_1\Phi^T+\frac{q}{q+r}\mathbf{I}
\end{aligned}
\tag{B.4}
$$

Then, update the covariance matrix $\mathbf{P}_1$,

$$
\begin{aligned}
\mathbf{P}_1=&(\mathbf{I}-\mathbf{G}_1)\mathbf{P}_1^-\\
=&\Phi\left[\frac{r}{q+r}\tilde{\mathbf{P}}_1^--\frac{r}{q+r}\tilde{\mathbf{G}}_1\Phi^T\Phi\tilde{\mathbf{P}}_1^--\frac{qr}{q+r}\tilde{\mathbf{G}}_1\right]\Phi^T\\
&+\left(q-\frac{q^2}{q+r}\right)\mathbf{I}\\
=&\Phi\tilde{\mathbf{P}}_1\Phi+\frac{qr}{q+r}\mathbf{I}
\end{aligned}
\tag{B.5}
$$

Finally, update μ_1

$$
\begin{aligned}
\mu_1=&\mu_1^-+\mathbf{G}_1(\phi(\mathbf{y}_1)-\mu_1^-)\\
=&\Phi\mathbf{a}_1+\left(\frac{r}{q+r}\Phi\tilde{\mathbf{G}}_1\Phi^T+\frac{q}{q+r}\mathbf{I}\right)(\varphi(\mathbf{y}_1)-\Phi\mathbf{a}_1)\\
=&\Phi\left[\left(\frac{r}{q+r}\mathbf{I}-\frac{r}{q+r}\tilde{\mathbf{G}}_1\Phi^T\Phi\right)\mathbf{a}_1+\frac{r}{q+r}\tilde{\mathbf{G}}_1\Phi^T\varphi(\mathbf{y}_1)\right]\\
&+\frac{q}{q+r}\varphi(\mathbf{y}_1)\\
=&\Phi\mathbf{b}_1+\frac{q}{q+r}\varphi(\mathbf{y}_1)
\end{aligned}
\tag{B.6}
$$

Here, we finished proving Theorem 4.2.1 and 4.2.2 for the first iteration, then we need to prove them for iteration $i > 1$. Suppose at iteration $i-1$, Theorem 4.2.1 and 4.2.2 are both satisfied, then at ith iteration, we have

$$\begin{aligned}\mu_i^- =&\mathbf{F}_{i-1}\mu_{i-1}\\ =&\Phi(\mathbf{K}+\zeta m\mathbf{I}_m)^{-1}\Upsilon^T\left[\Phi\mathbf{b}_{i-1}+\frac{q}{q+r}\varphi(\mathbf{y}_{i-1})\right]\\ =&\Phi\mathbf{a}_i\end{aligned} \tag{B.7}$$

and

$$\begin{aligned}\mathbf{P}_i^- =&\mathbf{F}_{i-1}\mathbf{P}_{i-1}\mathbf{F}_{i-1}^T+q\mathbf{I}\\ =&\mathbf{F}_{i-1}\left[\Phi\tilde{\mathbf{P}}_{i-1}\Phi^T+\frac{qr}{q+r}\mathbf{I}\right]\mathbf{F}_{i-1}^T+q\mathbf{I}\\ =&\Phi\tilde{\mathbf{P}}_i^-\Phi^T+q\mathbf{I}\end{aligned} \tag{B.8}$$

The other equations for iteration $i > 1$ can be computed using (B.2) to (B.6).

Appendix C

Proof of Theorem 4.3.1

Proof: Suppose $\mathbf{e}_j = [0, 0, ..., 1, ..., 0]^T (1 \leq j \leq k)$ is a canonical basis with k dimensions. And suppose $(\mathbf{y}_i^0, \mathbf{z}_i^0)$ is quantized to $(\mathbf{y}_j^q, \mathbf{z}_j^q)$, $\varphi(\mathbf{y}_i^0)$ and $\varphi(\mathbf{z}_i^0)$ are quantized to $\Upsilon_Q \mathbf{e}_j$ and $\Phi_Q \mathbf{e}_j$. We define $c[i] = j$ for convenience. The feature matrices Υ and Φ are quantized to $\Upsilon_Q \mathbf{E}$ and $Phi_Q \mathbf{E}$, where $\mathbf{E} = [\mathbf{e}_{c[1]}, ..., \mathbf{e}_{c[m]}]$. Then bring Υ and Φ into (4.24), we have

$$\mathbf{F}_Q = \Phi_Q \mathbf{E} (\mathbf{E}^T \mathbf{K}_Q \mathbf{E} + \zeta m \mathbf{I})^{-1} \mathbf{E}^T \Upsilon_Q^T \tag{C.1}$$

Then by matrix inversion lemma [9], we have

$$\begin{aligned} \mathbf{F}_Q =& \Phi_Q \mathbf{E}\mathbf{E}^T (\mathbf{K}_Q \mathbf{E}\mathbf{E}^T + \zeta m \mathbf{I})^{-1} \Upsilon_Q^T \\ =& \Phi_Q \Lambda (\mathbf{K}_Q \Lambda + \zeta m \mathbf{I})^{-1} \Upsilon_Q^T \end{aligned} \tag{C.2}$$

where $\Lambda = \mathbf{E}\mathbf{E}^T$ is a diagonal matrix, the jth diagonal component λ_j^q is the number of samples in jth bin from the kernel K-means algorithm.

Bibliography

[1] ANDERSON, B., AND MOORE, J. *Optimal Filtering*. Prentice-Hall, 1979.

[2] ARONSZAJN, N. Theory of reproducing kernels. *Trans. Amer. Math. Soc. 68* (1950), 337–404.

[3] BAKIR, G. H., WESTON, J., AND SCHÖLKOPF, B. Learning of find pre-images. *Advances in Neural Information Processing Systems 16* (2004), 449–456.

[4] CORTES, C., AND VAPNIK, V. Support-vector networks. In *Machine Learning* (1995), pp. 273–297.

[5] FUKUMIZU, K., BACH, F. R., AND JORDAN, M. I. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *JMLR 5* (2004), 73–99.

[6] FUKUMIZU, K., GRETTOU, A., SUN, X., AND SCHÖLKOPF, B. Kernel measures of conditional dependence. *Adv. Neural Inf. Process. Sys.*, 20 (2008), 489–496.

[7] FUKUMIZU, K., SONG, L., AND GRETTON, A. Kernel bayes' rule. *Adv. Neural Inf. Process. Syst.* (2011).

[8] GHOSHAL, D. P., GOPALAKRISHNAN, K., AND MICHALSKA, H. Algebraic parameter estimation using kernel representation of linear systems. *IFAC World Congress, 2017. 20th World Congress* (2017).

[9] H, H., AND SEARLE, S. On deriving the inverse of a sum of matrices. *SIAM Rev 23*, 1 (1981), 53–60.

[10] HAYKIN, S. *Kalman Filtering and Networks*. Wiley, New York, NY, USA, 2001.

[11] HENDERSON, H., AND SEARLE, S. On deriving the inverse of a sum of matrices. *SLAM Rev. 23*, 1 (1981), 53–60.

[12] IKEDA, K. Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system. *Opt. Commun*, 30 (1979), 257–261.

[13] ILLINGWORTH, V. *The Penguin dictionary of physics*. London: Penguin Books, 1991.

[14] JAZWINSKI, A. *Stochastic Process and Filtering Theory*. New York: Academic Press, 1970.

[15] JOHN, A. Estimation for siso lti systems using differential invariance. Master's thesis, McGill University, 2017.

[16] JULIER, S. J., AND UHLMANN, J. K. A new approach for filtering nonlinear systems. *Proceedings of the American Control Conference* (1995), 1628–1632.

[17] JULIER, S. J., AND UHLMANN, J. K. A general method for approximating nonlinear transformations of probability distributions. *Technical Report, RRG, Department of Engineering Science, University of Oxford* (1996).

[18] JULIER, S. J., AND UHLMANN, J. K. A new extension of the kalman filter to nonlinear systems. *Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls* (1997).

[19] KALMAN, R. E. A new approach to linear filtering and prediction problem. *Transactions of the ASME, Ser. D, Journal of Basic Engineering 82* (1960), 34–45.

[20] LIN, C. Support vector machines and kernel methods: Status and challenges. `https://www.csie.ntu.edu.tw/~cjlin/talks/kuleuven_svm.pdf`, 2013.

[21] LIU, W., PRINCIPE, J. C., AND HAYKIN, S. *Kernel Adaptive Filtering: A Comprehensive Introduction*. Wiley, 2010.

[22] MACQUEEN, J. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability 1* (1967), 281–297.

[23] MAYBECK, P. *Stochastic Models, Estimation and Control, vol. 1*. New York: Academic Press, 1979.

[24] MUANDET, K., FUKUMIZU, K., SRIPERUMBUDUR, B., AND SCHÖLKOPF, B. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning 10*, 1-2 (2017), 1–141.

[25] PEARSON, K. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11 (1901), 559–572.

[26] ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review* (1958), 65–386.

[27] SCHÖLKOPF, B., HERBRICH, R., AND SMOLA, A. J. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computer Learning Theory* (London, U.K., 2001), pp. 416–426.

[28] SCHÖLKOPF, B., SMOLA, A., AND MÜLLER, K. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput 10*, 5 (1998), 1299–1319.

[29] SONG, L., HUANG, J., SMOLA, A., AND FUKUMINZU, K. Hilbert space embeddings of conditional distributions with applications to dynamical systems. *Proc. 26th Int. Conf. Mach. Learn.* (2009), 961–968.

[30] TREES, H. V. *Detection, Estimation, and Modulation Theory, Part I*. Wiley, 1968.

[31] UHLMAN, J. K. Algorithms for multiple target tracking. *American Scientist 80(2)* (1992), 128–141.

[32] WAN, E. A., AND MERWE, R. V. The unscented kalman filter for nonlinear estimation. *Proc. of IEEE Symposium 2000 (AS-SPCC), Lake Louise, Alberta, Canada* (2000).

[33] WAN, E. A., MERWE, R. V., AND NELSO, A. T. Dual estimation and the unscented transformation. *Advances in Neural Information Processing Systems 12* (2000), 666–672.

[34] WELCH, G., AND BISHOP, G. An introduction to the kalman filter. *Technical report, UNC-CH Computer Science Technical Report 95041* (1995).

[35] ZHU, P., CHEN, B., AND PRINCIPE, J. C. Learning nonlinear generative models of time series with a kalman filter in rkhs, 2014.