Moving-horizon and kernel-based Extended Kalman filter for Non-Linear Estimation

Nikhil Jaiyam

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada

July 2021

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Engineering.

Abstract

This thesis presents a kernel-based parameter and state estimator built on various implementations of Recursive Least Squares estimators. The project represented a system using kernels in Reproducing Kernel Hilbert Spaces(RKHS) and co-variance propagation. Subsequently, a parameter estimation problem is solved using stochastic multiple regression and Generalized Least Squares with co-variance weighting applied to resolve high noise. Additionally, multiple integrals of the kernel for noise rejection and for multiple regression of a high order non-linear system are developed. This recursive method is then extended to a Moving-Horizon batch estimator with an adaptive window length. Furthermore, shortcomings of all the methods implemented are discussed to improve the method into a robust kernel-based extended Kalman filter algorithm for joint state and parameter estimation of Non-linear systems.

Résumé

Cette thèse présente un estimateur de paramètres et d'état basé sur des noyaux, construit sur diverses implémentations d'estimateurs de type Recursive Least Squares. Le projet a représenté un système utilisant des noyaux dans des espaces de Hilbert à noyaux reproducteurs (RKHS) et la propagation de co-variance. Ensuite, un problème d'estimation de paramètre est résolu en utilisant la régression multiple stochastique et les moindres carrés généralisés avec une pondération de co-variance appliquée pour résoudre le bruit élevé. De plus, des intégrales multiples du noyau pour la réjection du bruit et pour la régression multiple d'un système non linéaire d'ordre élevé sont développées. Cette méthode récursive est ensuite étendue à un estimateur par lots à horizon mobile avec une longueur de fenêtre adaptative. De plus, les défauts de toutes les méthodes mises en œuvre sont discutés afin d'améliorer la méthode en un algorithme robuste de filtre de Kalman étendu basé sur le noyau pour l'estimation conjointe de l'état et des paramètres des systèmes non linéaires.

Acknowledgment

I would like to thank my supervisor Dr.Hannah Michalska, for providing me the opportunity to work under her supervision and be part of the ongoing innovative research work. She has been of immense help in guiding me through tough times even during a global pandemic and allowing me the flexibility to better adapt to my research interests while keeping in check my financial health.

This wouldn't have been possible without guidance and collaboration by my fellow labmates Surya kumar Devarajan, Santosh Devapati, Shantanil Bagchi, Luis Medrano Del Rosal and Adarsh Mahesh Kumar. Their remarkable energy and integrity were a constant source of inspiration. Furthermore, I am deeply grateful to Karan shah, Rhythm sharma, Trinneta Bhojwani, Devanshi Bhargava and Hatice Irmaklı (and all my other friends)for their constant support, patience and being my family away from home.

Last but not least, I would like to convey my deepest thanks to my mother, my father and my sisters for their constant support throughout my life, both financially and emotionally and for their motivation to achieve the best. I am grateful to them for constantly supporting my endeavours and standing by me during all times.

Preface

This is to declare that the work presented in this document was completed and carried out by Nikhil Jaiyam in collaboration with Shantanil Bagchi, Luis Medrano Del Rosal and Adarsh Mahesh Kumar under the guidance of Professor Hannah Michalska. It builds on the efforts of Debarshi Patanjali Ghoshal, Surya kumar Devarajan and Santosh Devapati in the research group who carried out the parameter estimation for linear systems by least-squares and recursive least-squares.
The theoretical background for the RKHS approaches is based on the research notes by Professor Hannah Michalska, which is duly acknowledged. The specific algorithm for moving horizon, adaptive length moving horizon and Kernel based Extended Kalman filter were developed and investigated by myself as presented in this thesis.

Contents

List of Figures

List of Tables

List of Acronyms

RKHS	Reproducing Kernel Hilbert Space
KS	Kolmogorov Smirnov
LTI	Linear Time Invariant
RMSD	Root Mean Square Difference
SISO	Single Input Single Output
BLUE	Best Linear Unbiased Estimator
OLS	Ordinary Least Squares
RLS	Recursive Least Squares
GLS	Generalised Least Squares
IV	Instrument Variable

Chapter 1

Introduction

In mathematical modeling, we seek to uncover general laws and principles that govern the behavior under investigation. As these laws and principles are not directly observable, they are formulated in terms of hypotheses. Such hypotheses about the structure and inner working of the behavioral process of interest are stated in terms of parametric families of probability distributions called models. The goal of modeling is to deduce the form of the underlying process by testing the viability of such models. Once a model is identified or in some cases approximated when no prior knowledge of the model structure is available to the user, along with some properties of its parameters (order, linearity, time variance...etc), and data have been collected (which may also be preceded with some noise reject steps), one is in a position to evaluate its goodness of fit, that is, how well it fits the observed data.

Goodness of fit is assessed by finding parameter values of a model that best fits the data. This fit is usually governed by an appropriate cost function that is to be minimized with respect to the parameters to be estimated and other constraints to be kept in check—a procedure called parameter estimation. The problem of estimator design for an observable system has been extensively dealt with by mathematicians, especially, Kalman and Luenberger [10], [11], [12], [13].

1.1 State and parameter estimation

Recently, new non-asymptotic methods evolved for state and parameter estimation on finite time intervals [14], comprising a specific set of methods known as algebraic estimation methods which provide theoretical guarantees of convergence. Algebraic estimation is based on repeated differentiation of the measured system output from generating a data set by running the system for a given time and then sampling from said data. The original algebraic differentiator was based on the truncated Taylor series signal approximation and attributed its properties to the introduction of an algebraic annihilator of the Taylor series coefficients up to any desired order. The operation of differentiation is expressed in terms of the operation of integration. Noise attenuation is attempted by repeated integration since the operation of integration acts as a low pass filter and shaping of the annihilator functions used to eliminate the effect of the initial conditions [15] [16] [17]. However, this method requires frequent re-initialization when used forward in time, and its noise rejection properties are characterized as non-standard [18]. These methods are sensitive to measurement noise and lose their convergence guarantees over a longer time.

A novel estimation scheme for the state and its time derivatives by repeated integration is implemented using the knowledge of the system characteristic equation in [19]. In this approach, the state equations are replaced with an output reproducing property on an arbitrary time interval, which follows directly from the knowledge of the system characteristic equation. Such a behavioral model is derived from the differential in-variance, which is characteristic to the system [20]. This eliminates the need for initial conditions, and the mathematical model is in the form of a homogeneous Fredholm integral equation of the second kind with a Hilbert Schmidt kernel [21] [22]. The mathematical interpretation of the behavioral model in a RKHS allows extracting signal and its time derivatives from output measurement, despite the presence of noise [21]. The double-sided kernel approach is discussed in chapter 2.

Algebraic state and parameter estimation of linear systems based on a special construction of a forward-backward kernel representation of linear differential invariants are extended to handle large noise in output measurement [1] [19] [23] [24] [25] . The parameter estimation is solved by stochastic regression. The regression model does not satisfy the Gauss-Markov theorem's assumptions in that the random regressor is correlated with the regression error, which is also heteroskedastic. These complications do not impede achieving high accuracy of estimation. This approach when employed on a recursive basis can be extended to estimating parameters of a noisy signal online either in batches or sequential data and further extended into the non-linear domain of systems.

1.2 Thesis contributions

Here we build on the work on the existing framework surrounding double-sided kernel based parameter and state estimation [19]. The contribution of this thesis can be summarized as follows;

- A kernel is defined to estimate the parameters, order 3 , the output trajectory and the derivatives of the system linear up to order 3, which is then extended to an nth order system.
- An Ordinary Least Squares based multiple regression algorithm is described for estimation.
- A Recursive Least Squares estimation algorithm is presented with various sampling strategies for linear systems.
- The recursive approach is extended to non-linear systems for linear fitting.
- An adaptive length moving horizon estimator is designed over the RLS to implement a local linearization of the non-linear signals.
- The moving horizon estimation implementation is extended to a kalman filter.

1.3 Thesis summary: objectives and organization

Generalized least squares (GLS) works well with heteroskedasticity as it takes into account the inequality of variance in the observations. Recursive Generalized Least Squares provides an efficient way to update the approximate solution of least squares and has faster parameter convergence. The primary objective of this Master's thesis is to develop a forward-backward kernel based state and parameter estimation using multiple regression equations to compensate for the heteroskedasticity. The Recursive Generalized Least Squares is used for regression and it employs inverse covariance weighting [19].
It is also shown how the higher-order kernels can be used in the reconstruction and estimation of lower-order homogeneous LTI systems. and when used in a recursive manner over short horizons, this nature of convergence in parameters can be extended to non-linear systems without much prior knowledge of the original system even in the presence of high gaussian noise. [26]

The objective of this thesis are stated as follows:

- Understand the essence of the mathematical background behind algebraic state and parameter estimation when used in conjunction with reproducing kernels(RKHS);
- Study and understand previous contributions of the students who worked with Professor Michalska on the non-asymptotic joint state and parameter estimation.
- Study and experiment with the relationship between system complexity, noise, linearity and horizon length with accuracy of estimation with kernel based RLS.
- Investigate estimation of state and parameters of non-linear systems in the presence of noise and minimal knowledge of the system using kernel based RLS.
- Develop and investigate Moving horizon estimation of Non-linear systems
- Extend the Moving horizon estimator to Kernel based Extended Kalman Filter.
- Analyze the performance of the developed methods with previous implementations of kernel based methods.

This thesis is structured as follows:

Chapter 1 provides a brief introduction to algebraic state and parameter estimation and extension to non-linear systems as the central topic in this thesis.

Chapter 2 studies the works of [1], [20] & [23] and focuses on the derivation and understanding of the double-sided kernels for third order homogeneous SISO LTI systems.

Chapter 3 discusses the kernel based Recursive Least Squares algorithm applied to Linear systems and derives the covariance matrix for our kernel.

Chapter 4 talks about order identification, local linearization and implementing kernel-RLS to non-Linear systems. This chapter also explores a moving window approach to non-linear estimation.

Chapter 5 develops an adaptive length moving horizon estimate, motivation for discretization and the subsequent upgrade to Kalman filters, understanding the extended Kalman filter and methodology of our kernel based extended Kalman filter along with results.

Chapter 6 includes brief remarks and possible future directions of work.

Chapter 2

A double-sided Kernel in SISO LTI Representation

As mentioned in the introduction, algebraic estimation has different advantages where the framework is first laid in [8] and [9].The theory for the problem considered in this thesis was initially briefly presented in [13]. It employs forward and backward integration and Cauchy formula for multiple integrals to convert a high order differential equation, that embodies a system invariant, into an integral form with no singularities at the boundaries of the observation window. In any observation window, the output reproducing form of system invariance can then be used to characterize system trajectories. An equivalent system representation takes the form of a subspace of an RKHS Hilbert space - a fact that can be used in denoisification of measured output [16]. The paper considers only single-input single-output systems and it states the problem as follows.

2.1 Development of double-sided Kernel [1]

2.1.1 General n^{th} order SISO LTI system:

Consider a general n^{th} order, strictly proper and minimal SISO LTI system in state space form evolving on a given finite time interval [a,b]:

$$\dot{x} = Ax + bu; \quad y = c^T x; \quad x \in R^n \tag{2.1}$$

where the system matrix A is in canonical form,

$$\begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-1} \end{bmatrix} \tag{2.2}$$

with matching dimensions of the system matrices and the characteristic equation

$$\lambda^n + a_{n-1}\lambda^{n-1} + ... + a_1\lambda + a_0 = 0 \tag{2.3}$$

The input-output equation for system (2.1) becomes

$$y^{(n)}(t) + a_{n-1}y^{(n-1)}(t) + ... + a_1 y^{(1)}(t) + a_0 y(t) = -b_{n-1}u^{(n-1)}(t) - ... - b_0 u(t) \tag{2.4}$$

where -$b_i, i = 0, ...n-1$ are the coefficients of the polynomial in the numerator of the rational transfer function for (2.1).

The estimation problem is stated as follows. Given an arbitrary finite inteval of time [a,b], suppose that:

- The dimension of the state vector of the LTI system is known apriori alongside with the system input function $u(t)$ with its derivatives $u^{(i)}(t), i = 1, ..., n-1$ for $t \in [a, b]$.
- The output of the system is observed as a single realization of a "continuous" measurement process $y_M(t) := y(t) + \eta(t), t \in [a, b]$ in which η denotes additive white Gaussian noise with unknown intensity (variance) σ^2.

The parameter estimation of a homogeneous system can be viewed as the identification of a differential invariant $\mathcal{I}$ ($\mathcal{I} \equiv 0$) which does not change under the action of the system dynamics

$$\begin{aligned} &\mathcal{I}(t, y(t), y^{(1)}(t), \cdots, y^{(n)}(t)) \\ &\qquad = y^{(n)}(t) + a_{n-1}y^{(n-1)}(t) + \cdots + a_0 y(t) \ ; \ t \geq 0 \end{aligned} \tag{2.5}$$

The problem with singularity at $t = 0$, encountered in [27] in the integral system representation is eliminated in [1] using two-sided forward-backward integration which leads to an alternative integral system representation of (2.5) in a reproducing kernel Hilbert space (RKHS).
The following theorem describing such system representation has been stated and proved.

Theorem 1. *There exist Hilbert-Schmidt kernels K_{DS}, K^i_{DS}, $i = 1, \cdots n-1$, such that the output function y of (2.5) is reproduced on any given interval $[a, b]$ in accordance with the action of the evaluation functional*

$$y(t) = \int_a^b K_{DS}(t,\tau)y(\tau)\ \mathrm{d}\tau\ ;\ \forall t \in [a,b] \tag{2.6}$$

and the derivatives of the output $y^{(1)}, \cdots y^{(n-1)}$ can be computed recursively by way of output integration, so that for $i = 1, \cdots n-1$ and for all $t \in [a, b]$:

$$y^{(i)}(t) = \sum_{k=0}^{i-1} b_k(t)y^{(k)}(t) + \int_a^b K^i_{DS}(t,\tau)y(\tau)\ \mathrm{d}\tau \tag{2.7}$$

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

The kernels of Theorem 1 have the following expression for a system of order n,

$$\begin{aligned} K_{F,y}(n,t,\tau) = &\sum_{j=1}^{n}(-1)^{j+1}\begin{pmatrix} n \\ j \end{pmatrix}\frac{n!(t-\tau)^{j-1}(\tau-a)^{n-j}}{(n-j)!(j-1)!} \\ &+ \sum_{i=0}^{n-1} a_i \sum_{j=0}^{i}(-1)^{j+1}\begin{pmatrix} i \\ j \end{pmatrix}\frac{n!(t-\tau)^{n-i+j-1}(\tau-a)^{n-j}}{(n-j)!(n-i+j-1)!} \end{aligned} \tag{2.8}$$

$$\begin{aligned} K_{B,y}(n,t,\tau) = &\sum_{j=1}^{n}\begin{pmatrix} n \\ j \end{pmatrix}\frac{n!(t-\tau)^{j-1}(b-\tau)^{n-j}}{(n-j)!(j-1)!} \\ &+ \sum_{i=0}^{n-1} a_i \sum_{j=0}^{i}\begin{pmatrix} i \\ j \end{pmatrix}\frac{n!(t-\tau)^{n-i+j-1}(b-\tau)^{n-j}}{(n-j)!(n-i+j-1)!} \end{aligned} \tag{2.9}$$

$$K_{F,u}(n,t,\tau) = \sum_{i=0}^{n-1} b_i \sum_{j=0}^{i} (-1)^{j+1} \begin{pmatrix} i \\ j \end{pmatrix} \frac{n!(t-\tau)^{n-i+j-1}(\tau-a)^{n-j}}{(n-j)!(n-i+j-1)!} \quad (2.10)$$

$$K_{B,u}(n,t,\tau) = \sum_{i=0}^{n-1} b_i \sum_{j=0}^{i} \begin{pmatrix} i \\ j \end{pmatrix} \frac{n!(t-\tau)^{n-i+j-1}(b-\tau)^{n-j}}{(n-j)!(n-i+j-1)!} \quad (2.11)$$

2.2 Kernel development for a third order system [2]

The double-sided kernel for a general third order characteristic polynomial is derived in this section

$$y^{(3)}(t) + a_2 y^{(2)}(t) + a_1 y^{(1)}(t) + a_0 y(t) = 0 \quad (2.12)$$

on an interval $[a, b]$. By multiplying(2.12) with $(\xi - a)^3$ and $(b - \zeta)^3$ we get

$$(\xi - a)^3 y^{(3)}(t) + a_2(\xi - a)^3 y^{(2)}(t) + a_1(\xi - a)^3 y^{(1)}(t) + a_0(\xi - a)^3 y(t) = 0 \quad (2.13)$$

$$(b - \zeta)^3 y^{(3)}(t) + a_2(b - \zeta)^3 y^{(2)}(t) + a_1(b - \zeta)^3 y^{(1)}(t) + a_0(b - \zeta)^3 y(t) = 0 \quad (2.14)$$

Integrate (2.13) and (2.14) thrice on the interval $[a, a + \tau]$ and $[b - \sigma, b]$, meaning we would integrate the (2.12) in the forward direction during the interval $[a, a + \tau]$ and in the backward direction during the interval $[b, b - \sigma]$.

Now consider that we integrate the first term (2.13) once,

$$\int_a^{a+\tau} (\xi - a)^3 y^{(3)}(\xi)\,\mathrm{d}\xi$$

$$\begin{aligned}
&= (\xi - a)^3 y^{(2)}(\xi)\,\big|_a^{a+\tau} - \int_a^{a+\tau} 3(\xi - a)^2 y^{(2)}(\xi)\,\mathrm{d}\xi \\
&= \tau^3 y^{(2)}(a+\tau) - \left[3(\xi - a)^2 y^{(1)}(\xi)\,\big|_a^{a+\tau} - \int_a^{a+\tau} 6(\xi - a) y^{(1)}(\xi)\mathrm{d}\xi\right] \\
&= \tau^3 y^{(2)}(a+\tau) - 3\tau^2 y^{(1)}(a+\tau) + 6(\xi - a) y^{(1)}(\xi)\,\big|_a^{a+\tau} - \int_a^{a+\tau} 6y(\xi)\,\mathrm{d}\xi \\
&= \tau^3 y^{(2)}(a+\tau) - 3\tau^2 y^{(1)}(a+\tau) + 6\tau y(a+\tau) - \int_a^{a+\tau} 6y(\xi)\,\mathrm{d}\xi \qquad (2.15)
\end{aligned}$$

Integrating the same for a second time, we get the upper limit on the integral to be a 'dummy variable', by setting $\xi' = a + \tau$,
$\tau^3 y^{(2)}(a+\tau)$ becomes $(\xi' - a)^3 y^{(2)}(\xi')$
$3\tau^2 y^{(1)}(a+\tau)$ is now $3(\xi' - a)^2 y^{(1)}(\xi')$
$6\tau y(a+\tau)$ becomes $6(\xi' - a) y(\xi')$

Integrating (2.15) for the second time with the above changes,

$$\int_a^{a+\tau}\int_a^{\xi'} (\xi - a)^3 y^{(3)}(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'$$

$$= \int_a^{a+\tau} (\xi' - a)^3 y^{(2)}(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau} 3(\xi' - a)^2 y^{(1)}(\xi')\,\mathrm{d}\xi' + \int_a^{a+\tau} 6(\xi' - a) y(\xi')\,\mathrm{d}\xi'$$

$$
\begin{aligned}
&- \int_a^{a+\tau}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi' \\
&= (\xi'-a)^3 y^{(1)}(\xi')\,|_a^{a+\tau} - \int_a^{a+\tau} 3(\xi'-a)^2 y^{(1)}(\xi')\,\mathrm{d}\xi' - \Bigg[3(\xi'-a)^2 y(\xi')\,|_a^{a+\tau} \\
&\quad - \int_a^{a+\tau} 6(\xi'-a)y(\xi')\,\mathrm{d}\xi'\Bigg] + \int_a^{a+\tau} 6(\xi'-a)y(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi' \\
&= \tau^3 y^{(1)}(a+\tau) - \Bigg[3(\xi'-a)^2 y(\xi')\,|_a^{a+\tau} - \int_a^{a+\tau} 6(\xi'-a)y(\xi')\,\mathrm{d}\xi'\Bigg] \\
&\quad - 3\tau^2 y(a+\tau) + \int_a^{a+\tau} 12(\xi'-a)y(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi' \\
&= \tau^3 y^{(1)}(a+\tau) - 6\tau^2 y(a+\tau) + \int_a^{a+\tau} 18(\xi'-a)y(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'
\end{aligned}
\tag{2.16}
$$

As in the previous step, the upper limit is again a 'dummy variable' and now we set $\xi'' = a+\tau$. Integrating again,

$$
\begin{aligned}
&\int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} (\xi-a)^3 y^{(3)}(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi'' \\
&\quad = \int_a^{a+\tau} (\xi''-a)^3 y^{(1)}(\xi'')\,\mathrm{d}\xi'' - \int_a^{a+\tau} 6(\xi''-a)^2 y(\xi'')\,\mathrm{d}\xi'' + \int_a^{a+\tau}\int_a^{\xi''} 18(\xi'-a)y(\xi')\,\mathrm{d}\xi'\mathrm{d}\xi'' \\
&\qquad - \int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi'' \\
&\quad = \tau^3 y(a+\tau) - \int_a^{a+\tau} 3(\xi''-a)^2 y(\xi'')\,\mathrm{d}\xi'' - \int_a^{a+\tau} 6(\xi''-a)^2 y(\xi'')\,\mathrm{d}\xi''
\end{aligned}
$$

$$+\int_a^{a+\tau}\int_a^{\xi''} 18(\xi'-a)y(\xi')\,\mathrm{d}\xi'\mathrm{d}\xi'' - \int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi''$$

$$= \tau^3 y(a+\tau) - \int_a^{a+\tau} 9(\xi''-a)^2 y(\xi'')\,\mathrm{d}\xi'' + \int_a^{a+\tau}\int_a^{\xi''} 18(\xi'-a)y(\xi')\,\mathrm{d}\xi'\mathrm{d}\xi''$$

$$- \int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} 6y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi'' \qquad (2.17)$$

Now repeating the above procedure for the second term in (2.13) we get the following steps,

$$\int_a^{a+\tau} a_2(\xi-a)^3 y^{(2)}(\xi)\,\mathrm{d}\xi$$

$$= a_2(\xi-a)^3 y^{(1)}(\xi)\,\Big|_a^{a+\tau} - \int_a^{a+\tau} 3a_2(\xi-a)^2 y^{(1)}(\xi)\,\mathrm{d}\xi$$

$$= a_2\tau^3 y^{(1)}(a+\tau) - \left[3a_2(\xi-a)^2 y(\xi)\,\Big|_a^{a+\tau} - \int_a^{a+\tau} 6a_2(\xi-a)y(\xi)\,\mathrm{d}\xi\right]$$

$$= a_2\tau^3 y^{(1)}(a+\tau) - 3a_2\tau^2 y(a+\tau) + \int_a^{a+\tau} 6a_2(\xi-a)y(\xi)\,\mathrm{d}\xi \qquad (2.18)$$

Introducing ‘dummy variable’ and integrating (2.18) again,

$$\int_a^{a+\tau}\int_a^{\xi'} a_2(\xi-a)^3 y^{(2)}(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'$$

$$= \int_a^{a+\tau} a_2(\xi'-a)^3 y^{(1)}(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau} 3a_2(\xi'-a)^2 y(\xi')\,\mathrm{d}\xi' + \int_a^{a+\tau}\int_a^{\xi'} 6a_2(\xi-a)y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'$$

$$= a_2(\xi'-a)^3 y(\xi')\,\Big|_a^{a+\tau} - \int_a^{a+\tau} 3a_2(\xi'-a)^2 y(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau} 3a_2(\xi'-a)^2 y(\xi')\mathrm{d}\xi'$$

$$+\int_a^{a+\tau}\int_a^{\xi'} 6a_2(\xi-a)y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'$$

$$=\tau^3 a_2(a+\tau)-\int_a^{a+\tau} 6a_2(\xi'-a)^2y(\xi')\,\mathrm{d}\xi'+\int_a^{a+\tau}\int_a^{\xi'} 6a_2(\xi-a)y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'$$

finally, integrating for the third time,

$$\int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} a_2(\xi-a)^3y^{(2)}(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi''$$

$$=\int_a^{a+\tau} a_2(\xi''-a)^3y(\xi'')\,\mathrm{d}\xi''-\int_a^{a+\tau}\int_a^{\xi''} 6a_2(\xi'-a)^2y(\xi')\,\mathrm{d}\xi'\mathrm{d}\xi''$$

$$+\int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} 6a_2(\xi-a)y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi'' \tag{2.19}$$

Similar steps are repeated for the third term in (2.13) ,

$$\int_a^{a+\tau} a_1(\xi-a)^3y^{(1)}(\xi)\,\mathrm{d}\xi$$

$$=a_1(\xi-a)^3y(\xi)\;|_a^{a+\tau}-\int_a^{a+\tau} 3a_1(\xi-a)^2y(\xi)\,\mathrm{d}\xi \tag{2.20}$$

$$=a_1\tau^3y(a+\tau)-\int_a^{a+\tau} 3a_1(\xi-a)^2y(\xi)\,\mathrm{d}\xi \tag{2.21}$$

Integrating (2.20) twice we get,

$$\int_a^{a+\tau}\int_a^{\xi'} a_1(\xi-a)^3 y^{(1)}(\xi)\,\mathrm{d}\xi\mathrm{d}\xi' = \int_a^{a+\tau} a_1(\xi'-a)^3 y(\xi')\,\mathrm{d}\xi' - \int_a^{a+\tau}\int_a^{\xi'} 3a_1(\xi-a)^2 y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi' \tag{2.22}$$

and finally,

$$\begin{aligned}\int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} a_1(\xi-a)^3 y^{(1)}(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi'' &= \int_a^{a+\tau}\int_a^{\xi''} a_1(\xi'-a)^3 y(\xi')\,\mathrm{d}\xi' \\ &\quad - \int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} 3a_1(\xi-a)^2 y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi''\end{aligned} \tag{2.23}$$

Similarly for the last term of the equation (2.13)

$$\int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} a_0(\xi-a)^3 y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi'' \tag{2.24}$$

arranging (2.17) to (2.24) together, we end up

$$\begin{aligned}-\tau^3 y(a+\tau) = &\int_a^{a+\tau} \Big[-9(\xi''-a)^2 + a_2(\xi''-a)^3\Big] y(\xi'')\,\mathrm{d}\xi'' \\ &+ \int_a^{a+\tau}\int_a^{\xi''} \Big[+18(\xi'-a) - 6a_2(\xi'-a)^2 + a_1(\xi'-a)^3\Big] y(\xi')\,\mathrm{d}\xi'\mathrm{d}\xi'' \\ &+ \int_a^{a+\tau}\int_a^{\xi''}\int_a^{\xi'} \Big[-6 + 6a_2(\xi-a) - 3a_1(\xi-a)^2 + a_0(\xi-a)^3\Big] y(\xi)\,\mathrm{d}\xi\mathrm{d}\xi'\mathrm{d}\xi''\end{aligned} \tag{2.25}$$

This can be further simplified by recalling Cauchy formula for repeated integration. Let f

be a continuous function on the real line, then the nth repeated integral of f based at a.

$$f^{(-n)}(x) = \int_a^x \int_a^{\sigma_1} \cdots \int_a^{\sigma_{n-1}} f(\sigma_n) \mathrm{d}\sigma_n \cdots \mathrm{d}\sigma_2 \mathrm{d}\sigma_1 \tag{2.26}$$

is given by single integration

$$f^{(-n)}(x) = \frac{1}{(n-1)!} \int_a^x (x-t)^{n-1} f(t) \mathrm{d}t \tag{2.27}$$

let $a + \tau = t$, $\tau = t - a$ and applying the above for repeated integration on (2.25),we get,

$$\begin{aligned}
&-(t-a)^3 y(t) \\
&\quad = \int_a^t \Big[-9(\tau-a)^2 + a_2(\tau-a)^3 \Big] y(\tau)\, \mathrm{d}\tau \\
&\qquad + \int_a^t (t-\tau) \Big[18(\tau-a) - 6a_2(\tau-a)^2 + a_1(\tau-a)^3 \Big] y(\tau)\, \mathrm{d}\tau \\
&\qquad + \frac{1}{2} \int_a^t (t-\tau)^2 \Big[-6 + 6a_2(\tau-a) - 3a_1(\tau-a)^2 + a_0(\tau-a)^3 \Big] y(\tau)\, \mathrm{d}\tau \\
&\quad \triangleq \int_a^t K_F(t,\tau) y(\tau)\, \mathrm{d}\tau
\end{aligned} \tag{2.28}$$

with $K_F(t,\tau)$ defined by,

$$\begin{aligned}
K_F(t,\tau) \triangleq \Big[-9(\tau-a)^2 + a_2(\tau-a)^3 \Big] + (t-\tau) \Big[18(\tau-a) - 6a_2(\tau-a)^2 + a_1(\tau-a)^3 \Big] \\
+ (t-\tau)^2 \Big[-6 + 6a_2(\tau-a) - 3a_1(\tau-a)^2 + a_0(\tau-a)^3 \Big]
\end{aligned} \tag{2.29}$$

Consider the equation (2.14). Integrating the first term in (2.14) once,

$$\int_{b-\sigma}^{b} (b-\zeta)^3 y^{(3)}(\zeta)\,\mathrm{d}\zeta$$

$$= (b-\zeta)^3 y^{(2)}(\zeta)\,|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 3(b-\zeta)^2 y^{(2)}(\zeta)\,\mathrm{d}\zeta$$

$$= -\sigma^3 y^{(2)}(b-\sigma) + \left[3(b-\zeta)^2 y^{(1)}(\zeta)\,|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 6(b-\zeta) y^{(1)}(\zeta), \mathrm{d}\zeta\right] \tag{2.30}$$

$$= -\sigma^3 y^{(2)}(b-\sigma) - 3\sigma^2 y^{(1)}(b-\sigma) + 6(b-\zeta)y^{(1)}(\zeta)\,|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 6y(\zeta)\,\mathrm{d}\zeta$$

$$= -\sigma^3 y^{(2)}(b-\sigma) - 3\sigma^2 y^{(1)}(b-\sigma) - 6\sigma y(b-\sigma) + \int_{b-\sigma}^{b} 6y(\zeta)\,\mathrm{d}\zeta \tag{2.31}$$

Following the earlier steps i.e, introducing a 'dummy variable' and setting $\zeta' = b - \sigma$,

$$
\begin{aligned}
&\int_{b-\sigma}^{b}\int_{b}^{\zeta'}(b-\zeta)^3 y^{(3)}(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \\
&\quad= \int_{b-\sigma}^{b}(b-\zeta')^3 y^{(2)}(\zeta')\,\mathrm{d}\zeta' + \int_{b-\sigma}^{b} 3(b-\zeta')^2 y^{(1)}(\zeta')\,\mathrm{d}\zeta' + \int_{b-\sigma}^{b} 6(b-\zeta')y(\zeta')\,\mathrm{d}\zeta' \\
&\qquad - \int_{b-\sigma}^{b}\int_{b}^{\zeta'} 6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \\
&\quad= (b-\zeta')^3 y^{(1)}(\zeta')\mid_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 3(b-\zeta')^2 y^{(1)}(\zeta')\,\mathrm{d}\zeta' + \Bigg[3(b-\zeta')^2 y(\zeta')\mid_{b-\sigma}^{b} \\
&\qquad + \int_{b-\sigma}^{b} 6(b-\zeta')y(\zeta')\,\mathrm{d}\zeta'\Bigg] + \int_{b-\sigma}^{b} 6(b-\zeta')y(\zeta')\,\mathrm{d}\zeta' + \int_{b}^{b-\sigma}\int_{b}^{\zeta'} 6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \\
&\quad= -\sigma^3 y^{(1)}(b-\sigma) + \Bigg[3(b-\zeta')^2 y(\zeta')\mid_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 6(b-\zeta')y(\zeta')\,\mathrm{d}\zeta'\Bigg] \\
&\qquad - 3\sigma^2 y(b-\sigma) - \int_{b}^{b-\sigma} 12(b-\zeta')y(\zeta')\,\mathrm{d}\zeta' + \int_{b}^{b-\sigma}\int_{b}^{\zeta'} 6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'
\end{aligned}
$$

$$
\begin{aligned}
&= -\sigma^3 y^{(1)}(b-\sigma) + \Bigg[-3\sigma^2 y(b-\sigma) + \int\limits_{b-\sigma}^{b} 6(b-\zeta')y(\zeta')\,\mathrm{d}\zeta' \Bigg] \\
&\qquad - 3\sigma^2 y(b-\sigma) - \int\limits_{b}^{b-\sigma} 12(b-\zeta')y(\zeta')\,\mathrm{d}\zeta' + \int\limits_{b}^{b-\sigma}\int\limits_{b}^{\zeta'} 6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \\
&= -\sigma^3 y^{(1)}(b-\sigma) - 6\sigma^2 y(b-\sigma) - \int\limits_{b}^{b-\sigma} 18(b-\zeta')y(\zeta')\,\mathrm{d}\zeta' \\
&\qquad + \int\limits_{b}^{b-\sigma}\int\limits_{b}^{\zeta'} 6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'
\end{aligned}
\tag{2.32}
$$

Again setting a 'dummy variable' , $\zeta'' = b - \sigma$.Integrate (2.32) third time,

$$\int_{b-\sigma}^{b}\int_{b}^{\zeta''}\int_{b}^{\zeta'}(b-\zeta)^3 y^{(3)}(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''$$

$$= \int_{b-\sigma}^{b}(b-\zeta'')^3 y^{(1)}(\zeta'')\,\mathrm{d}\zeta'' + \int_{b-\sigma}^{b}6(b-\zeta'')^2 y(\zeta'')\,\mathrm{d}\zeta'' - \int_{b}^{b-\sigma}\int_{b}^{\zeta''}18(b-\zeta')y(\zeta')\,\mathrm{d}\zeta'\mathrm{d}\zeta''$$

$$- \int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'}6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''$$

$$= \left[(b-\zeta'')^3 y(\zeta'')\,\Big|_{b-\sigma}^{b} + \int_{b-\sigma}^{b}3(b-\zeta'')^2 y(\zeta'')\,\mathrm{d}\zeta''\right] - \int_{b}^{b-\sigma}6(b-\zeta'')^2 y(\zeta'')\,\mathrm{d}\zeta''$$

$$- \int_{b}^{b-\sigma}\int_{b}^{\zeta''}18(b-\zeta')y(\zeta')\,\mathrm{d}\zeta'\mathrm{d}\zeta'' - \int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'}6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''$$

$$= -\sigma^3 y(b-\sigma) - \int_{b}^{b-\sigma}9(b-\zeta'')^2 y(\zeta'')\,\mathrm{d}\zeta'' - \int_{b}^{b-\sigma}\int_{b}^{\zeta''}18(b-\zeta')y(\zeta')\,\mathrm{d}\zeta'\mathrm{d}\zeta''$$

$$- \int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'}6y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \tag{2.33}$$

The above steps are repeated for all the terms in (2.14),

$$\int_{b-\sigma}^{b} a_2(b-\zeta)^3 y^{(2)}(\zeta)\,\mathrm{d}\zeta$$

$$= a_2(b-\zeta)^3 y^{(1)}(\zeta)\,|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 3a_2(b-\zeta)^2 y^{(1)}(\zeta)\,\mathrm{d}\zeta$$

$$= -a_2\sigma^3 y^{(1)}(b-\sigma) + \left[3a_2(b-\zeta)^2 y(\zeta)\,|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 6a_2(b-\zeta)y(\zeta)\,\mathrm{d}\zeta\right]$$

$$= -a_2\sigma^3 y^{(1)}(b-\sigma) - 3a_2\sigma^2 y(b-\sigma) - \int_{b}^{b-\sigma} 6a_2(b-\zeta)y(\zeta)\,\mathrm{d}\zeta \qquad (2.34)$$

Using 'dummy variable' and integrating (2.34),

$$\int_{b-\sigma}^{b}\int_{b}^{\zeta'} a_2(b-\zeta)^3 y^{(2)}(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'$$

$$= \int_{b-\sigma}^{b} a_2(b-\zeta')^3 y^{(1)}(\zeta')\,\mathrm{d}\zeta' + \int_{b-\sigma}^{b} 3a_2(b-\zeta')^2 y(\zeta')\,\mathrm{d}\zeta x - \int_{b-\sigma}^{b}\int_{b}^{\zeta'} 6a_2(b-\sigma)y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'$$

$$= a_2(b-\zeta')^3 y(\zeta')\,|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 3a_2(b-\zeta')^2 y(\zeta')\,\mathrm{d}\zeta' - \int_{b}^{b-\sigma} 3a_2(b-\zeta')^2 y(\zeta')$$

$$- \int_{b-\sigma}^{b}\int_{b}^{\zeta'} 6a_2(b-\zeta)y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'$$

$$= -\sigma^3 a_2(b-\sigma) - \int_{b}^{b-\sigma} 6a_2(b-\zeta')^2 y(\zeta')\,\mathrm{d}\zeta' - \int_{b-\sigma}^{b}\int_{b}^{\zeta'} 6a_2(b-\zeta)y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \qquad (2.35)$$

and for the third time,

$$\int_{b-\sigma}^{b}\int_{b}^{\zeta''}\int_{b}^{\zeta'} a_2(b-\zeta-a)^3 y^{(2)}(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''$$

$$\begin{aligned}
&= \int_{b-\sigma}^{b} a_2(b-\zeta'')^3 y(\zeta'')\,\mathrm{d}\zeta'' + \int_{b-\sigma}^{b}\int_{b}^{\zeta''} 6a_2(b-\zeta')^2 y(\zeta')\,\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\quad + \int_{b-\sigma}^{b}\int_{b}^{\zeta''}\int_{b}^{\zeta'} 6a_2(b-\zeta)y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&= -\int_{b}^{b-\sigma} a_2(b-\zeta'')^3 y(\zeta'')\,\mathrm{d}\zeta'' - \int_{b}^{b-\sigma}\int_{b}^{\zeta''} 6a_2(b-\zeta')^2 y(\zeta')\,\mathrm{d}\zeta'\mathrm{d}\zeta'' - \\
&\quad \int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'} 6a_2(b-\zeta)y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''
\end{aligned} \tag{2.36}$$

$$\int_{b-\sigma}^{b} a_1(b-\zeta)^3 y^{(1)}(\zeta)\,\mathrm{d}\zeta$$

$$\begin{aligned}
&= a_1(b-\zeta)^3 y(\zeta)\,\Big|_{b-\sigma}^{b} + \int_{b-\sigma}^{b} 3a_1(b-\zeta)^2 y^{(2)}(\zeta)\,\mathrm{d}\zeta \\
&= -a_1\sigma^3 y(b-\sigma) - \int_{b}^{b-\sigma} 3a_1(b-\zeta)^2 y^{(2)}(\zeta)\,\mathrm{d}\zeta
\end{aligned} \tag{2.37}$$

Integrating (2.37) second time,

$$\int_{b-\sigma}^{b}\int_{b}^{\zeta'} a_1(b-\zeta)^3 y^{(1)}(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' = \int_{b-\sigma}^{b} a_1(b-\zeta)^3 y(\zeta')\,\mathrm{d}\zeta' - \int_{b-\sigma}^{b}\int_{b}^{\zeta'} 3a_1(b-\zeta)^2 y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \tag{2.38}$$

$$= -\int_{b}^{b-\sigma} a_1(b-\zeta)^3 y(\zeta')\,\mathrm{d}\zeta' + \int_{b}^{b-\sigma}\int_{b}^{\zeta'} 3a_1(b-\zeta)^2 y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta' \tag{2.39}$$

Integrating (2.38) the final time,

$$\begin{aligned}
&\int_{b-\sigma}^{b}\int_{b}^{\zeta''}\int_{b}^{\zeta'} a_1(b-\zeta)^3 y^{(1)}(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\qquad = \int_{b-\sigma}^{b}\int_{b}^{\zeta''} a_1(b-\zeta')^3 y(\zeta')\,\mathrm{d}\zeta' + \int_{b-\sigma}^{b}\int_{b}^{\zeta''}\int_{b}^{\zeta'} 3a_1(b-\zeta)^2 y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&\qquad = -\int_{b}^{b-\sigma}\int_{b}^{\zeta''} a_1(b-\zeta')^3 y(\zeta')\,\mathrm{d}\zeta' - \int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'} 3a_1(b-\zeta)^2 y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''
\end{aligned} \tag{2.40}$$

the last term in (2.14)

$$\int_{b-\sigma}^{b}\int_{b}^{\zeta''}\int_{b}^{\zeta'} a_0(b-\zeta)^3 y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' = -\int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'} a_0(b-\zeta)^3 y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta'' \tag{2.41}$$

all the integrated terms from (2.32) to (2.41) together form:

$$\begin{aligned}
\sigma^3 y(b-\sigma) = &\int_{b}^{b-\sigma}\Big[-9(b-\zeta'')^2 - a_2(b-\zeta'')^3\Big]y(\zeta'')\,\mathrm{d}\zeta'' \\
&+\int_{b}^{b-\sigma}\int_{b}^{\zeta''}\Big[-18(b-\zeta') - 6a_2(b-\zeta')^2 - a_1(b-\zeta')^3\Big]y(\zeta')\,\mathrm{d}\zeta'\mathrm{d}\zeta'' \\
&+\int_{b}^{b-\sigma}\int_{b}^{\zeta''}\int_{b}^{\zeta'}\Big[-6 - 6a_2(b-\zeta) - 3a_1(b-\zeta)^2 - a_0(b-\zeta)^3\Big]y(\zeta)\,\mathrm{d}\zeta\mathrm{d}\zeta'\mathrm{d}\zeta''
\end{aligned} \tag{2.42}$$

let $b-\sigma = t$ and $\sigma = b-t$, applying the Cauchy formula for repeated integration as stated earlier,

$$
\begin{aligned}
(b-t)^3 y(t) = &\int_b^t \Big[-9(b-\sigma)^2 - a_2(b-\sigma)^3 \Big] y(\sigma)\,\mathrm{d}\sigma \\
&+ \int_b^t (t-\sigma)\Big[-18(b-\sigma) - 6a_2(b-\sigma)^2 - a_1(b-\sigma)^3 \Big] y(\sigma)\,\mathrm{d}\sigma \\
&+ \frac{1}{2}\int_b^t (t-\sigma)^2\Big[-6 - 6a_2(b-\sigma) - 3a_1(b-\sigma)^2 - a_0(b-\sigma)^3 \Big] y(\sigma)\,\mathrm{d}\sigma
\end{aligned}
\tag{2.43}
$$

flipping the limits on integrals (2.43) can be written as,

$$
\begin{aligned}
(b-t)^3 y(t) = &-\int_t^b \Big[-9(b-\tau)^2 - a_2(b-\tau)^3 \Big] y(\tau)\,\mathrm{d}\tau \\
&- \int_t^b (t-\tau)\Big[-18(b-\tau) - 6a_2(b-\tau)^2 - a_1(b-\tau)^3 \Big] y(\tau)\,\mathrm{d}\tau \\
&- \frac{1}{2}\int_t^b (t-\tau)^2\Big[-6 - 6a_2(b-\tau) - 3a_1(b-\tau)^2 - a_0(b-\tau)^3 \Big] y(\tau)\,\mathrm{d}\tau \\
\triangleq &\int_b^t K_B(t,\tau) y(\tau)\mathrm{d}\tau
\end{aligned}
\tag{2.44}
$$

where $K_B(t,\tau)$ defined as,

$$
\begin{aligned}
K_B(t,\tau) \triangleq &\Big[9(b-\tau)^2 + a_2(b-\tau)^3 \Big] \\
&+ (t-\tau)\Big[18(b-\tau) + 6a_2(b-\tau)^2 + a_1(b-\tau)^3 \Big] \\
&+ \frac{1}{2}(t-\tau)^2\Big[6 + 6a_2(b-\tau) + 3a_1(b-\tau)^2 + a_0(b-\tau)^3 \Big]
\end{aligned}
\tag{2.45}
$$

' *The forward*' & '*backward*' partial kernels are redefined as

$$\begin{aligned} K_F(t,\tau) \triangleq \mu(\tau-a)&\Bigg[9(\tau-a)^2 - a_2(\tau-a)^3\Bigg] \\ &+ (t-\tau)\Bigg[-18(\tau-a) + 6a_2(\tau-a)^2 - a_1(\tau-a)^3\Bigg] \\ &+ \frac{1}{2}(t-\tau)^2\Bigg[6 - 6a_2(\tau-a) + 3a_1(\tau-a)^2 - a_0(\tau-a)^3\Bigg] \end{aligned} \tag{2.46}$$

$$\begin{aligned} K_B(t,\tau) \triangleq \mu(b-\tau)&\Bigg[9(b-\tau)^2 + a_2(b-\tau)^3\Bigg] \\ &+ (t-\tau)\Bigg[18(b-\tau) + 6a_2(b-\tau)^2 + a_1(b-\tau)^3\Bigg] \\ &+ \frac{1}{2}(t-\tau)^2\Bigg[6 + 6a_2(b-\tau) + 3a_1(b-\tau)^2 + a_0(b-\tau)^3\Bigg] \end{aligned} \tag{2.47}$$

where

$$\mu(\tau-a) = \begin{cases} 1 & : \tau \geq a \\ 0 & : \tau < a \end{cases}$$

and

$$\mu(b-\tau) = \begin{cases} 1 & : \tau \leq b \\ 0 & : \tau > b \end{cases}$$

The equations (2.46) and (2.47) can now be expressed simply as :

$$(t-a)^3 y(t) = \int_a^t K_F(t,\tau) y(\tau)\,\mathrm{d}\tau \tag{2.48}$$

$$(b-t)^3 y(t) = \int_t^b K_B(t,\tau) y(\tau)\,\mathrm{d}\tau \tag{2.49}$$

Hence,

$$K_{DS}(t,\tau) \triangleq \begin{cases} K_F(t,\tau) & : \tau \leq t \\ K_B(t,\tau) & : \tau > t \end{cases} \tag{2.50}$$

Combining (2.49) and (2.50) and dividing both sides by $[(t-a)^3+(b-t)^3]$ results in:

$$y(t) = \frac{1}{[(t-a)^3+(b-t)^3]}\int\limits_a^b K_{DS}(t,\tau)y(\tau)\,\mathrm{d}\tau \tag{2.51}$$

The recursive expressions of the derivatives can be derived by following similar steps as in the derivation of K_{DS}. In order to obtain the expression for $y^{(1)}(t)$ (2.13) and (2.14) are integrated twice.

$$\begin{aligned}(t-a)^3y^{(1)}(t) &= 6(t-a)^2y(t) - a_2(t-a)^3y(t)\\ &+ \int_t^a \Big[-18(\tau-a)+6a_2(\tau-a)^2-a_1(\tau-a)^3\Big]y(\tau)\mathrm{d}\tau\\ &+ \int_t^a (t-\tau)\Big[6-6a_2(\tau-a)^2-3a_1(\tau-a)^2-a_0(\tau-a)^3\Big]y(\tau)\mathrm{d}\tau\end{aligned} \tag{2.52}$$

and

$$\begin{aligned}(b-t)^3y^{(1)}(t) &= -6(b-t)^2y(t) - a_2(b-t)^3y(t)\\ &+ \int_t^b \Big[18(b-\tau)+6a_2(b-\tau)^2+a_1(b-\tau)^3\Big]y(\tau)\mathrm{d}\tau\\ &+ \int_t^b (t-\tau)\Big[6+6a_2(b-\tau)^2+3a_1(b-\tau)^2+a_0(b-\tau)^3\Big]y(\tau)\mathrm{d}\tau\end{aligned} \tag{2.53}$$

$y^{(1)}(t)$ is obtained by adding (2.52) and (2.53) and dividing by $[(t-a)^3+(b-t)^3]$.
For $y^{(2)}(t)$, (2.13) and (2.14) are integrated only once , hence resulting in

$$\begin{aligned}(t-a)^3y^{(2)}(t) &= 3(t-a)^2y^{(1)}(t) - a_2(t-a)^3y^{(1)}(t)\\ &- 6(t-a)y(t) + 3a_2(t-a)^2y(t) - a_1(t-a)^3y(t)\\ &+ \int_t^a \Big[6-6a_2(\tau-a)+3a_1(\tau-a)^2-a_0(\tau-a)^3\Big]y(\tau)\mathrm{d}\tau\end{aligned} \tag{2.54}$$

and

$$\begin{aligned}(b-t)^3y^{(2)}(t) = &-3(b-t)^2y^{(1)}(t) - a_2(b-t)^3y^{(1)}(t)\\ &-6(b-t)y(t) - 3a_2(b-t)^2y(t) - a_1(b-t)^3y(t)\\ &+\int_t^b \left[6 + 6a_2(b-\tau) + 3a_1(b-\tau)^2 + a_0(b-\tau)^3\right] y(\tau)\mathrm{d}\tau\end{aligned} \tag{2.55}$$

$y^{(2)}(t)$ is also obtained by following similar steps as above and hence adding (2.54) and (2.55) and also dividing by $[(t-a)^3 + (b-t)^3]$.

2.2.1 Kernels for a 4th order homogeneous LTI system

The kernels for the fourth-order system are developed by substituting $n = 4$ in the equations (2.8) and (2.9) by using Theorem 1.

$$\begin{aligned}K_{F,y}(4,t,\tau) = &16(\tau-a)^3 - 72(t-\tau)(\tau-a)^2\\ &+48(t-\tau)^2(\tau-a) - 4(t-\tau)^3 + a_0[-\frac{1}{6}(t-\tau)^3(\tau-a)^4]\\ &+a_1[-\frac{1}{2}(t-\tau)^2(\tau-a)^4 + \frac{2}{3}(t-\tau)^3(\tau-a)^3]\\ &+a_2[-(t-\tau)(\tau-a)^4 + 4(t-\tau)^2(\tau-a)^3 - 2(t-\tau)^3(\tau-a)^2]\\ &+a_3[-(\tau-a)^4 + 12(t-\tau)(\tau-a)^3 - 18(t-\tau)^2(\tau-a)^2\\ &+4(t-\tau)^3(\tau-a)]\end{aligned} \tag{2.56}$$

$$\begin{aligned}K_{B,y}(4,t,\tau) = &16(b-\tau)^3 + 72(t-\tau)(b-\tau)^2\\ &+48(t-\tau)^2(b-\tau) + 4(t-\tau)^3 + a_0[\frac{1}{6}(t-\tau)^3(b-\tau)^4]\\ &+a_1[\frac{1}{2}(t-\tau)^2(b-\tau)^4 + \frac{2}{3}(t-\tau)^3(b-\tau)^3]\\ &+a_2[(t-\tau)(b-\tau)^4 + 4(t-\tau)^2(b-\tau)^3 + 2(t-\tau)^3(b-\tau)^2]\\ &+a_3[(b-\tau)^4 + 12(t-\tau)(b-\tau)^3 + 18(t-\tau)^2(b-\tau)^2\\ &+4(t-\tau)^3(b-\tau)]\end{aligned} \tag{2.57}$$

with

$$K_{DS,y}(4,t,\tau) \triangleq \begin{cases} K_{F,y}(4,t,\tau), & \text{for } \tau \leq t \\ K_{B,y}(4,t,\tau), & \text{for } \tau > t \end{cases} \tag{2.58}$$

And thus we have $y(t)$ in terms of kernel representation as:

$$y(t) = \int_a^b K_{DS,y}(t,\tau)y(\tau)\ d\tau \tag{2.59}$$

Consider a general third order system as below:

$$y^{(3)}(t) + a_2 y^{(2)}(t) + a_1 y^{(1)}(t) + a_0 y(t) = 0 \tag{2.60}$$

If the equation (2.60) is differentiated, we will have:

$$y^{(4)}(t) + a_2 y^{(3)}(t) + a_1 y^{(2)}(t) + a_0 y^{(1)}(t) = 0 \tag{2.61}$$

Now, we write the two systems in this format (both as $4th$ order systems with different coefficients) :

$$b_4 y^{(4)}(t) + b_3 y^{(3)}(t) + b_2 y^{(2)}(t) + b_1 y^{(1)}(t) + b_0 y(t) = 0 \tag{2.62}$$

If (2.62) represents (2.60) then we would have to identify:

$$b_4 = 0, b_3 = 1, b_2 = a_2, b_1 = a_1, b_0 = a_0 \tag{2.63}$$

If (2.62) represents (2.61) then we would have to identify:

$$b_4 = 1, b_3 = a_3, b_2 = a_2, b_1 = a_1, b_0 = 0 \tag{2.64}$$

if a curve $y^*(t)$ fits (2.60) for $t \in [a, b]$, then it also fits (2.61) in the format of (2.62) with coefficients respectively in (2.63) and (2.64). This would indicate that y^* will be reproduced by a $3rd$ order kernel for (2.60) but also by a $4th$ order kernel using respective coefficients.

2.2.2 Kernels for a 5th order homogeneous LTI system

The kernels for the fifth-order system are developed by substituting $n = 5$ in the equations (2.8) and (2.9) by using Theorem 1.

$$\begin{aligned} K_{F,y}(5,t,\tau) = {} & 25(\tau-a)^4 - 200(t-\tau)(\tau-a)^3 + 300(t-\tau)^2(\tau-a)^2 \\ & -100(t-\tau)(t-\tau)^3 + 5(t-\tau)^4 + a_0[-\frac{1}{24}(t-\tau)^4(\tau-a)^5] \\ & +a_1[-\frac{1}{6}(t-\tau)^3(\tau-a)^5 + \frac{5}{24}(t-\tau)^4(\tau-a)^4] + a_2[-\frac{1}{2}(t-\tau)^2 \\ & (\tau-a)^5 + \frac{10}{6}(t-\tau)^3(\tau-a)^4 - \frac{20}{24}(t-\tau)^4(\tau-a)^3] \\ & +a_3[-(\tau-a)^5(t-\tau) + \frac{15}{2}(t-\tau)^2(\tau-a)^4 - 10(t-\tau)^3(\tau-a)^3 \\ & +\frac{60}{24}(t-\tau)^4(\tau-a)^2] + a_4[-\frac{120}{24}(\tau-a)(t-\tau)^4 + \frac{240}{6}(t-\tau)^3 \\ & (\tau-a)^2 - 60(t-\tau)^2(\tau-a)^3 + 20(t-\tau)(\tau-a)^4 - (\tau-a)^5] \end{aligned} \tag{2.65}$$

$$
\begin{aligned}
K_{B,y}(5,t,\tau) = {} & 25(b-\tau)^4 + 200(t-\tau)(b-\tau)^3 + 300(t-\tau)^2(b-\tau)^2 \\
& +100(t-\tau)(t-\tau)^3 + 5(t-\tau)^4 + a_0[\frac{1}{24}(t-\tau)^4(b-\tau)^5] \\
& +a_1[\frac{1}{6}(t-\tau)^3(b-\tau)^5 + \frac{5}{24}(t-\tau)^4(b-\tau)^4] + a_2[\frac{1}{2}(t-\tau)^2 \\
& (b-\tau)^5 + \frac{10}{6}(t-\tau)^3(b-\tau)^4 + \frac{20}{24}(t-\tau)^4(b-\tau)^3] \\
& +a_3[(b-\tau)^5(t-\tau) + \frac{15}{2}(t-\tau)^2(b-\tau)^4 + 10(t-\tau)^3(b-\tau)^3 \\
& +\frac{60}{24}(t-\tau)^4(b-\tau)^2] + a_4[-\frac{120}{24}(b-\tau)(t-\tau)^4 + \frac{240}{6}(t-\tau)^3 \\
& (b-\tau)^2 + 60(t-\tau)^2(b-\tau)^3 + 20(t-\tau)(b-\tau)^4 + (b-\tau)^5] \qquad (2.66)
\end{aligned}
$$

with

$$
K_{DS,y}(5,t,\tau) \triangleq \begin{cases} K_{F,y}(5,t,\tau), & \text{for } \tau \le t \\ K_{B,y}(5,t,\tau), & \text{for } \tau > t \end{cases} \qquad (2.67)
$$

(2.67) corresponds to the fifth-order kernels. To reproduce a third-order system from the fifth order kernels, we assume that a_0 and a_1 terms are zero, while the constant term, a_2, a_3 and a_4 constitute the third-order system.

The kernels defined above will be used in subsequent chapters to form multiple regression equations by integrating the reproducing property multiple times. The equations formed can then be solved by Recursive Least Squares method to jointly estimate state and parameter values of any constant parameter LTI system.

Chapter 3

Finite Interval Parameter and State estimation in LTI Systems Using kernel based Multiple Regression

In this chapter, the parameters are obtained by recursive generalized least squares involving multiple regression. By using the reproducing property of Theorem 1, the kernel representation in (2.48) is first integrated two times to deliver two additional linearly independent regression equations; this is to match the number of the parameters with the number of regression equations. A third-order system is considered with its unknown parameters a_0, a_1, a_2, strictly proper and minimal SISO LTI system in state space form evolving on a given finite time interval [a,b]:

$$\dot{x} = Ax + bu; \quad y = c^T x; \quad x \in R^n \tag{3.1}$$

where the system matrix A is in canonical form,

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_0 & -a_1 & -a_2 \end{bmatrix} \tag{3.2}$$

with matching dimensions of the system matrices and the characteristic equation

$$a_2\lambda^2 + a_1\lambda + a_0 = 0 \tag{3.3}$$

The forward kernel representation (2.48) for system (3.1) can be written simply as,

$$\alpha_a(t)y(t) \triangleq \int_a^t K_F(t,s)y(s)ds \tag{3.4}$$

and using the Cauchy formula for repeated integration on (3.4) yields

$$\frac{1}{(n-1)!}\int_a^t \alpha_a(s)(t-s)^{n-1}y(s)ds = \frac{1}{n!}\int_a^t (t-s)^n K_F(t,s)y(s)ds \quad n = 1,2 \tag{3.5}$$

From (2.29), we have $K_F(t,s)$ defined as,

$$\begin{aligned} K_F(t,s) = &\Big[9(s-a)^2 - a_2(s-a)^3\Big] \\ &+ (t-s)\Big[-18(s-a) + 6a_2(s-a)^2 - a_1(s-a)^3\Big] \\ &+ (t-s)^2\Big[6 - 6a_2(s-a) + 3a_1(s-a)^2 - a_0(s-a)^3\Big] \end{aligned} \tag{3.6}$$

Substituting (3.6) in (3.5), we get

$$
\begin{aligned}
&\frac{1}{(n-1)!}\int_a^t \alpha_a(s)(t-s)^{n-1}y(s)ds \\
&= \int_a^t \frac{(t-s)^n}{n!}\Big[9(s-a)^2 - a_2(s-a)^3\Big]y(s)ds \\
&+ \int_a^t \frac{(t-s)^{n+1}}{(n+1)!}\Big[-18(s-a) + 6a_2(s-a)^2 - a_1(s-a)^3\Big]y(s)ds \\
&+ \int_a^t \frac{(t-s)^{n+2}}{(n+2)!}\Big[6 - 6a_2(s-a) + 3a_1(s-a)^2 - a_0(s-a)^3\Big]y(s)ds
\end{aligned}
\tag{3.7}
$$

The equation (3.5) can be re-written as

$$
\frac{1}{(n-1)!}\int_a^t \alpha_a(s)(t-s)^{n-1}y(s)ds = \int_a^t K_F^n(t,s)y(s)ds \quad \text{for } n = 1, 2 \tag{3.8}
$$

where $K_F^n(t,s)$ is

$$
\begin{aligned}
K_F^n(t,s) &= \frac{(t-s)^n}{n!}\Big[9(s-a)^2 - a_2(s-a)^3\Big] \\
&+ \frac{(t-s)^{n+1}}{(n+1)!}\Big[-18(s-a) + 6a_2(s-a)^2 - a_1(s-a)^3\Big] \\
&+ \frac{(t-s)^{n+2}}{(n+2)!}\Big[6 - 6a_2(s-a) + 3a_1(s-a)^2 - a_0(s-a)^3\Big]
\end{aligned}
\tag{3.9}
$$

Note: equation (3.8) will be used to formulate the forward kernels for the additional two ($n = 1, 2$) regression equations besides (3.4) for the system (3.1). However, for a system with more number of parameters n can be further extended to obtain kernels.

A similar procedure is then used for the backward kernel representation.

$$
\alpha_b(t)y(t) = \int_t^b K_B(t,s)y(s)ds = -\int_b^t K_B(t,s)y(s)ds \tag{3.10}
$$

using the Cauchy formula for repeated integration on (3.10) yields

$$\frac{1}{(n-1)!}\int_t^b \alpha_b(s)(t-s)^{n-1}y(s)ds = -\frac{1}{n!}\int_t^b (t-s)^n K_B(t,s)y(s)ds \quad \text{for } n=1,2 \tag{3.11}$$

From (2.45), we have $K_B(t,s)$ defined as,

$$\begin{aligned} K_B(t,s) = &\Big[9(b-s)^2 + a_2(b-s)^3\Big] \\ &+ (t-s)\Big[18(b-s) + 6a_2(b-s)^2 + a_1(b-s)^3\Big] \\ &+ \frac{1}{2}(t-s)^2\Big[6 + 6a_2(b-s) + 3a_1(b-s)^2 + a_0(b-s)^3\Big] \end{aligned} \tag{3.12}$$

Substituting (3.12) in (3.11), we get

$$\begin{aligned} &\frac{1}{(n-1)!}\int_t^b \alpha_b(s)(t-s)^{n-1}y(s)ds \\ &= -\Bigg[\Bigg[\int_t^b \frac{(t-s)^n}{n!}\Big[9(b-s)^2 + a_2(b-s)^3\Big]y(s)ds \\ &+ \int_t^b \frac{(t-s)^{n+1}}{(n+1)!}\Big[18(b-s) + 6a_2(b-s)^2 + a_1(b-s)^3\Big]y(s)ds \\ &+ \int_t^b \frac{(t-s)^{n+2}}{(n+2)!}\Big[6 + 6a_2(b-s) + 3a_1(b-s)^2 + a_0(b-s)^3\Big]y(s)ds\Bigg] \end{aligned} \tag{3.13}$$

The equation (3.11) can be re-written as

$$\frac{1}{(n-1)!}\int_t^b \alpha_b(s)(t-s)^{n-1}y(s)ds = -\int_t^b K_B^n(t,s)y(s)ds \tag{3.14}$$

where $K_B^n(t,s)$ is

$$\begin{aligned} K_B^n(t,s) = \Bigg[& \frac{(t-s)^n}{n!}\Big[9(b-s)^2 + a_2(b-s)^3\Big] \\ & + \frac{(t-s)^{n+1}}{(n+1)!}\Big[18(b-s) + 6a_2(b-s)^2 + a_1(b-s)^3\Big] \\ & + \frac{(t-s)^{n+2}}{(n+2)!}\Big[6 + 6a_2(b-s) + 3a_1(b-s)^2 + a_0(b-s)^3\Big]\Bigg] \end{aligned} \tag{3.15}$$

Note: equation (3.14) will be used to formulate the backward kernels for the additional two ($n = 1, 2$) regression equations besides (3.10) for the system (3.1). However, for a system with more number of parameters n can be further extended to obtain more kernels.

Adding (3.4) and (3.10) yields

$$\alpha_a(t)y(t) + \alpha_b(t)y(t) = \int_a^t K_F(t,s)y(s)ds + \int_t^b K_B(t,s)y(s)ds \tag{3.16}$$

$$\alpha_a(t)y(t) + \alpha_b(t)y(t) = \int_a^b K_{DS}(t,s)y(s)ds \tag{3.17}$$

$$\alpha_{ab}(t)y(t) = \int_a^b K_{DS}(t,s)y(s)ds \tag{3.18}$$

with

$$K_{DS}(t,s) \triangleq \begin{cases} K_F(t,s) : s \le t \\ K_B(t,s) : s > t \end{cases} \tag{3.19}$$

and

$$\alpha_{ab}(t) = \alpha_a(t) + \alpha_b(t) \tag{3.20}$$

Note: equation (3.18) **represents the first regression equation(which is also the reproducing property) for our reference third order system** (3.1)**.**
adding (3.8) and (3.14) yields

$$\frac{1}{(n-1)!}\left[\int_a^t \alpha_a(s)(t-s)^{n-1}y(s)ds + \int_t^b \alpha_b(s)(t-s)^{n-1}y(s)ds\right] = \int_a^t K_F^n(t,s)y(s)ds + \int_t^b -K_B^n(t,s)y(s)ds \tag{3.21}$$

$$\frac{1}{(n-1)!}\left[\int_a^t \alpha_a(s)(t-s)^{n-1}y(s)ds + \int_t^b \alpha_b(s)(t-s)^{n-1}y(s)ds\right] = \int_a^b K_{DS}^n(t,s)y(s)ds \tag{3.22}$$

Similar to (3.18), we can write (3.22) as,

$$\frac{1}{(n-1)!}\left[\int_a^b \alpha_{ab}(s)(t-s)^{n-1}y(s)ds\right] = \int_a^b K_{DS}^n(t,s)y(s)ds \tag{3.23}$$

with

$$K_{DS}^n(t,s) \triangleq \begin{cases} K_F^n(t,s) : s \leq t \\ -K_B^n(t,s) : s > t \end{cases} \tag{3.24}$$

The kernels of Theorem(1) are linear in the unknown system coefficients β_i. Hence, the reproducing property for homogeneous systems is first re-written to bring out this fact while omitting the dependence of the kernels on n [3]. The additional linearly independent equations formed by repeated integration as in (3.23) with $n = 1, 2$ can also be represented. The equations (3.18) and (3.23) can be re-written as follows

$$\alpha_{ab}(t)y(t) = \sum_{i=0}^{n} \beta_i \int_a^b K_{DS}(t,s)y(s)ds \tag{3.25}$$

$$\int_a^b \alpha_{ab}(s)y(s)ds = \sum_{i=0}^{n} \beta_i \int_a^b K_{DS}^1(t,s)y(s)ds \quad \text{for } n=1 \tag{3.26}$$

$$\int_a^b \alpha_{ab}(s)(t-s)y(s)ds = \sum_{i=0}^{n} \beta_i \int_a^b K_{DS}^2(t,s)y(s)ds \quad \text{for } n=2 \tag{3.27}$$

where the K_{DS} and K_{DS}^n, $i = 0, ..., n$ are 'component kernels' of K_{DS} and K_{DS}^n that post-multiply the coefficients $\beta_i = a_i$, $i = 0, ..., n-1$, with $\beta_n = 1$ for convenience of notation. The output variable y becomes the measured output coinciding with the nominal output trajectory y_T.

With $\bar{a} := [a_0; \cdots; a_{n-1}]$ and $\beta := [a_0, \cdots, a_{n-1}, a_n] = [\bar{a}; 1]$, let $K_{DS_{k}(\bar{a})}(t, y_T)$ be row vectors with integral components

The equations (3.25),(3.26),(3.27) can be written as follows

$$\begin{bmatrix} \alpha_{ab}(s)y(s)ds - K_{DS(a_n)}(t,y_T) \\ \displaystyle\int_a^b \alpha_{ab}(s)(t-s)y(s)ds - K_{DS(a_n)}^1(t,y_T) \\ \displaystyle\frac{1}{2}\int_a^b \alpha_{ab}(s)(t-s)^2y(s)ds - K_{DS(a_n)}^2(t,y_T) \end{bmatrix} = \begin{bmatrix} K_{DS(\bar{a})}(t,y_T) \\ K_{DS(\bar{a})}^1(t,y_T) \\ K_{DS(\bar{a})}^2(t,y_T) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \tag{3.28}$$

Given distinct time instants $t_1, ..., t_N \in (a, b)$, here referred to as knots, the regression equation is re-written point-wise in the form of a matrix equation

$$Q(y_T) = P(y_T)\overline{a} \tag{3.29}$$

with

$$\overline{a} = [a_0, a_1, a_2`] \tag{3.30}$$

for a third order system.

The above mentioned regression equations are replicated N times and can be written in a matrix form as:

$$Q = \begin{bmatrix} q^1(t_1) \\ \vdots \\ q^1(t_N) \\ \vdots \\ q^3(t_1) \\ \vdots \\ q^3(t_N) \end{bmatrix} ; P = \begin{bmatrix} p_0^1(t_1) & \dots & p_{n-1}^1(t_1) \\ \vdots & \ddots & \vdots \\ p_0^1(t_N) & \dots & p_{n-1}^1(t_N) \\ \vdots & \ddots & \vdots \\ p_0^3(t_1) & \dots & p_{n-1}^3(t_1) \\ \vdots & \ddots & \vdots \\ p_0^3(t_N) & \dots & p_{n-1}^3(t_N) \end{bmatrix} \tag{3.31}$$

Where we rewrite (3.25), (3.26),(3.27) to apply least squares estimation as,

$$Q = \begin{bmatrix} q^1(t_i) \\ q^2(t_i) \\ q^3(t_i) \end{bmatrix} = \begin{bmatrix} \alpha_{ab}(t)y_T(t_i) - \beta_n \left[\int_a^b K_{DS(a_n)}(t_i, s)y(s)ds \right] \\ \int_a^b \alpha_{ab}(s)y(s)ds - \beta_n \left[\int_a^b K_{DS(a_n)}^1(t_i, s)y(s)ds \right] \\ \int_a^b \alpha_{ab}(s)(t_i - s)y(s)ds - \beta_n \left[\int_a^b K_{DS(a_n)}^2(t, s)y(s)ds \right] \end{bmatrix} \tag{3.32}$$

$$P = \begin{bmatrix} p^1(t_i) \\ p^2(t_i) \\ p^3(t_i) \end{bmatrix} = \begin{bmatrix} \int_a^b K_{DS(\bar{a})}(t_i, s)y(s)ds \\ \int_a^b K_{DS(\bar{a})}^1(t_i, s)y(s)ds \\ \int_a^b K_{DS(\bar{a})}^2(t_i, s)y(s)ds \end{bmatrix} \tag{3.33}$$

where $Q(y_T) \in \mathbb{R}^{Nk}$, $P(y_T) \in \mathbb{R}^{Nk} \times \mathbb{R}^n$ and $\bar{a} \in \mathbb{R}^n$ (n=3 for the third order system).The regression equation in (3.29) can be solved using least-squares error minimization provided adequate identifiability assumptions are met, and the output is measured without error.

3.1 Identifiability of homogeneous LTI systems from a single realization of a measured output [3] [4]

Identifiability of a homogeneous LTI system such as

$$\begin{aligned} &\dot{x}(t) = Ax(t); \quad y = Cx; \quad x \in \mathbb{R}^n \\ &x(0) = b \end{aligned} \tag{3.34}$$

from a single noise-free realization of its output trajectory y on the interval $[0, \infty)$ has been studied by Stanhope [28]. The identifiability condition stated in equivalent form:
Definition 2: Model (3.34) is globally identifiable from b if and only if the functional mapping $A \mapsto y(\cdot; A, b)$ is injective on $\mathbb{R}^n$ where $y(\cdot; A, b)$ denotes the output orbit of 3.34 through b.

Theorem 1. *Model (3.34) is globally identifiable from b if and only if the output orbit of (3.34) is not confined to a proper subspace of $\mathbb{R}^n$.*

The above criterion has limited use for reasons of practicality: it is difficult to verify computationally, pertains to infinite time horizons $[0, \infty)$, and most importantly, requires the output trajectory to be known exactly. Hence it suffices to invoke a practical version of identifiability as defined below.

Definition 1: Practical linear identifiability
The homogeneous system is practically linearly identifiable on $[a, b]$ with respect to a particular noisy discrete realization of the output measurement process, $y(t), t \in [a, b]$, if and only if there exist distinct knots $t_1, \cdots, t_N \in (a, b]$ which render $rankP(y_M) = n$. Any such output realization is then called *persistent*.

In the presence of large noise in system output, here assumed to be AWGN - white gaussian and additive, the regression equation (3.25),(3.26) and (3.27) is no longer valid as the reproducing property fails to hold along an inexact output trajectory. It must thus be suitably replaced leading to a regression problem. First, the output is a stochastic process on a probability space $L^2(\Omega, \mathcal{F}, \mathbb{P})$ where y_M is assumed to be adapted to the natural filtration of the standard Wiener process W on $[a, b]$ is

$$y_M(t, \omega) = y_T(t) + \sigma \dot{W}(t, \omega) \ ; \quad t \in [a, b] \tag{3.35}$$

where y_T signifies the true system output and $\sigma \dot{W}$ is the white noise process with constant variance σ^2.

The following facts are recalled from [29] [30]. For any smooth and compactly supported function $g = g(t)$ the generalized derivative $\dot{w}(t)$ of $w(t)$ (not obligatory differentiable function) is defined symbolically as

$$\int_0^\infty g(t)\dot{w}(t)dt = -\int_0^\infty \dot{g}(t)w(t)dt \tag{3.36}$$

For a smooth $w(t)$, when $\dot{w}(t)$ exists, the above formula is nothing but "integration by parts" formula.

The generalized derivative $\dot{W}(t)$ of Wiener process, defined similarly to integration by parts with a smooth g(t)

$$g(t)W_t = \int_0^t g(s)\dot{W}_s ds + \int_0^t \dot{g}(s)W_s ds \tag{3.37}$$

is called "Gaussian white noise".

Main properties of Gaussian white noise are as follows [31] [32]:

1. Expectation of white noise:
$$E[\dot{W}_t] \equiv 0 \tag{3.38}$$

2. Covariance function of white noise is a dirac function:
$$Cov[\dot{W}_t, \dot{W}_s] = E[\dot{W}_t \dot{W}_s] = \delta(t - s) \tag{3.39}$$

3. Variance:
$$Var[\dot{W}_t] = E[\dot{W}_t^2] = 1 \tag{3.40}$$

4. $\int_0^t g(s)\dot{W}_s ds$ is well defined, if only

$$\int_0^t g(s)^2 ds < \infty \tag{3.41}$$

5. Moreover, for any square integrable function g,

$$E\int_0^t g(s)\dot{W}_s ds = 0 \tag{3.42}$$

$$E\Big(\int_0^t g(s)\dot{W}_s ds\Big)^2 = \int_0^t g(s)^2 ds \tag{3.43}$$

Here δ is the delta Dirac distribution acting on square integrable functions as an evaluation functional:

$$\int_a^b g(s)\delta(t-s)ds = g(t), \quad t \in [a,b] \tag{3.44}$$

The equation (3.34) is rewritten again,

$$y_M(t,\omega) = y_T(t) + \sigma\dot{W}(t,\omega)\ ; \quad t \in [a,b] \tag{3.45}$$

This corresponds to a random kernel expression,

$$\int_a^b K_{DS,y}(t,s)y_M(s)\ d\tau = \sum_{i=0}^{n}\beta_i\int_a^b K_{DS(i),y}(t,s)y_M(s)ds \tag{3.46}$$

The above expression can be better-written w.r.t (3.45) as follows,

$$\int_a^b K_{DS,y}(t,s)y_M(s)\ d\tau = \int_a^b K_{DS,y}(t,s)y_T(s)\ ds + \int_a^b K_{DS,y}(t,s)\sigma\dot{W}(s) \tag{3.47}$$

Here, $\sigma\dot{W}(s)$ is the Wiener process with σ as intensity and W as the standard Brownian motion. This applies to all the independent regression equations (3.18) and (3.23). It follows that the following equalities are valid

$$\alpha_{ab}(t)y_M(t) = \int_a^b K_{DS,y}(t,s)y_M(s)ds + e_1(t) \tag{3.48}$$

$$\int_a^b \alpha_{ab}(s)y_M(s)ds = \int_a^b K^1_{DS,y}(t,s)y_M(s)ds + e_2(t) \tag{3.49}$$

$$\int_a^b \alpha_{ab}(s)(t-s)y_M(s)ds = \int_a^b K^2_{DS,y}(t,s)y_M(s)ds + e_3(t) \tag{3.50}$$

The above equations can also be written as

$$\alpha_{ab}(t)y_M(t) = \sum_{i=0}^{n} \beta_i \int_a^b K_{DS}(t,s)y_M(s)ds + e_1(t) \tag{3.51}$$

$$\int_a^b \alpha_{ab}(s)y_M(s)ds = \sum_{i=0}^{n} \beta_i \int_a^b K^1_{DS}(t,s)y_M(s)ds + e_2(t) \tag{3.52}$$

$$\int_a^b \alpha_{ab}(s)(t-s)y_M(s)ds = \sum_{i=0}^{n} \beta_i \int_a^b K^2_{DS}(t,s)y_M(s)ds + e_3(t) \tag{3.53}$$

The above equations can be represented in the matrix form as follows,

$$\begin{bmatrix} \alpha_{ab}(t)y_M(t) \\ \int_a^b \alpha_{ab}(s)y_M(s)ds \\ \int_a^b \alpha_{ab}(s)(t-s)y_M(s)ds \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n} \beta_i \int_a^b K_{DS}(t,s)y_M(s)ds \\ \sum_{i=0}^{n} \beta_i \int_a^b K^1_{DS}(t,s)y_M(s)ds \\ \sum_{i=0}^{n} \beta_i \int_a^b K^2_{DS}(t,s)y_M(s)ds \end{bmatrix} + \begin{bmatrix} e_1(t) \\ e_2(t) \\ e_3(t) \end{bmatrix} \tag{3.54}$$

Substituting (3.28) into (3.54), we get

$$\begin{bmatrix} \alpha_{ab}(t)y_M(t) - \beta_n \int_a^b K_{DS(a_n)}(t,s)y_M(s)ds \\ \int_a^b \alpha_{ab}(s)y_M(s)ds - \beta_n \int_a^b K^1_{DS(a_n)}(t,s)y_M(s)ds \\ \int_a^b \alpha_{ab}(s)(t-s)y_M(s)ds - \beta_n \int_a^b K^2_{DS(a_n)}(t,s)y_M(s)ds \end{bmatrix} = \begin{bmatrix} \int_a^b K_{DS(\bar{a})}(t,s)y_M(s)ds \\ \int_a^b K^1_{DS(\bar{a})}(t,s)y_M(s)ds \\ \int_a^b K^2_{DS(\bar{a})}(t,s)y_M(s)ds \end{bmatrix} \bar{a} + \begin{bmatrix} e_1(t) \\ e_2(t) \\ e_3(t) \end{bmatrix} \tag{3.55}$$

The equations (3.48), (3.49), (3.57) have the random regressor matrix

$$\begin{bmatrix} \int_a^b K_{DS(0),y}(t,s)y_M(s)ds, \cdots, \int_a^b K_{DS(a_n),y}(t,s)y_M(s)ds \\ \int_a^b K^1_{DS(0),y}(t,s)y_M(s)ds, \cdots, \int_a^b K^1_{DS(a_n),y}(t,s)y_M(s)ds \\ \int_a^b K^2_{DS(0),y}(t,s)y_M(s)ds, \cdots, \int_a^b K^2_{DS(a_n),y}(t,s)y_M(s)ds \end{bmatrix}^T \tag{3.56}$$

and the error terms are written as

$$e_1(t) := \alpha_{ab}(t)(\sigma\dot{W}(t)) - \int_a^b K_{DS,y}(t,s)\sigma\dot{W}(s)ds \tag{3.57}$$

$$e_2(t) := \int_a^b \alpha_{ab}(s)(\sigma\dot{W}(s))ds - \int_a^b K^1_{DS,y}(t,s)\sigma\dot{W}(s)ds \tag{3.58}$$

$$e_3(t) := \int_a^b \alpha_{ab}(s)(t-s)(\sigma\dot{W}(s))ds - \int_a^b K^2_{DS,y}(t,s)\sigma\dot{W}(s)ds \tag{3.59}$$

since y_T satisfies the reproducing property in the deterministic regression equation (3.25),(3.26) and (3.27) . It is noted that the random error variable e is dependent on the unknown system parameters $a_i, i = 0, \cdots, n-1$. It can be verified that the assumptions of Gauss-Markov Theorem are violated in the linear regression problem because the random regressor is correlated with a regression error, which additionally fails to be homoskedastic. The problem of heteroskedasticity is dealt next.

3.2 Heteroskedasticity [3]

Heteroskedasticity has serious consequences for the Ordinary Least Squares estimator. Despite the fact that the OLS estimator remains unbiased, the estimated regression error is wrong, while confidence intervals cannot be relied on. The best way to deal with heteroskedasticity is to employ Recursive Generalized Least Squares (GLS) as it employs inverse covariance weighting in the regression error minimization problem associated with (3.51), (3.52) and (3.53). Let $Q(y_M)$ and $P(y_M)$ as defined in the (3.32) be the matrices corresponding to N samples of the measurement process realization y_M at a batch of knots $t_1, \cdots, t_N$. Then the stochastic regression error vector is given by

$$e_n := [e_n(t_i), \cdots, e_n(t_N)]^T = Q(y_M) - P(y_M)\bar{a} \tag{3.60}$$

where $e_n(t_i)$ are as in (3.57), (3.58) and (3.59). The standard GLS regression error minimization for estimation of the parameter vector $\bar{a}$ is

$$\min_{\bar{a}} \left([Q(y_M) - P(y_M)\bar{a}]^T S[Q(y_M) - P(y_M)\bar{a})] \right) \tag{3.61}$$

$$\text{with} \quad S := [\mathrm{Cov}(e)]^{-1} \tag{3.62}$$

Applying the expectation operator to equations (3.34), (3.57), (3.58) and (3.59) and using the properties of the Gaussian noise yields

$$\mathrm{E}[y_M(t)] = \mathrm{E}[y_T(t)] + \mathrm{E}[\sigma \dot{W}(t)] = y_T(t) \quad t \in [a, b] \tag{3.63}$$

$$\mathrm{E}[e_1(t)] = \mathrm{E}\left[\alpha_{ab}(t)(\sigma \dot{W}(t))\right] - \mathrm{E}\left[\int_a^b K_{DS,y}(t,s)\sigma \dot{W}(s)ds\right] = 0 \tag{3.64}$$

$$\mathrm{E}[e_2(t)] = \mathrm{E}\left[\int_a^b \alpha_{ab}(s)(\sigma \dot{W}(s))ds\right] - \mathrm{E}\left[\int_a^b K^1_{DS,y}(t,s)\sigma \dot{W}(s)ds\right] = 0 \tag{3.65}$$

$$\mathrm{E}[e_3(t)] = \mathrm{E}\left[\int_a^b \alpha_{ab}(s)(t-s)(\sigma \dot{W}(s))ds\right] - \mathrm{E}\left[\int_a^b K^2_{DS,y}(t,s)\sigma \dot{W}(s)ds\right] = 0 \tag{3.66}$$

thus

$$\mathrm{Cov}(e) = \mathrm{E}\left[ee^T\right] \tag{3.67}$$

Following is the calculation of the error covariance matrix, which is used in the GLS algorithm.

3.3 Calculation of error covariance matrix needed for recursive GLS method

For simplicity of exposition, we will derive the the error covariance matrix with a third order kernel for a third order system (3.1). This method was subsequently extended for an n^{th} order system in [5].

3.3.1 Error covariance matrix for a third order system

The error terms e_1, e_2, and e_3 are specified in the equations (3.57), (3.58), and (3.59). Using the properties (3.38) - (3.44) of AWGN and recalling that the kernel functions are Hilbert-Schmidt, hence are square integrable, the covariance matrix is then calculated as follows.

$$
\begin{aligned}
&Cov[e_1(t_i), e_1(t_j)] = E[e_1(t_i)e_1(t_j)] \\
&= \sigma^2 E\Big[\Big[\alpha_{ab}(t_i)\dot{W}(t_i) - \int_a^b K_{DS}(t_i,s)\dot{W}(s)ds\Big]\Big[\alpha_{ab}(t_j)\dot{W}(t_j) - \int_a^b K_{DS}(t_j,\tau)\dot{W}(\tau)d\tau\Big]\Big]
\end{aligned} \tag{3.68}
$$

$$
\begin{aligned}
&= \sigma^2 E\Big[\alpha_{ab}(t_i)\alpha_{ab}(t_j)\dot{W}(t_i)\dot{W}(t_j)\Big] - \sigma^2 E\Big[\alpha_{ab}(t_i)\dot{W}(t_i)\int_a^b K_{DS}(t_j,\tau)\dot{W}(\tau)d\tau\Big] \\
&\quad - \sigma^2 E\Big[\alpha_{ab}(t_j)\dot{W}(t_j)\int_a^b K_{DS}(t_i,s)\dot{W}(s)ds\Big] \\
&\quad + \sigma^2 E\Big[\int_a^b\int_a^b K_{DS}(t_i,s)K_{DS}(t_j,\tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big]
\end{aligned} \tag{3.69}
$$

$$
\begin{aligned}
&= \sigma^2 E\Big[\alpha_{ab}(t_i)\alpha_{ab}(t_j)\dot{W}(t_i)\dot{W}(t_j)\Big] - \sigma^2 E\Big[\alpha_{ab}(t_i)\int_a^b K_{DS}(t_j,\tau)\dot{W}(t_i)\dot{W}(\tau)d\tau\Big] \\
&\quad - \sigma^2 E\Big[\alpha_{ab}(t_j)\int_a^b K_{DS}(t_i,s)\dot{W}(t_j)\dot{W}(s)\Big]ds\Big] \\
&\quad + \sigma^2 E\Big[\int_a^b\int_a^b K_{DS}(t_i,s)K_{DS}(t_j,\tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big]
\end{aligned} \tag{3.70}
$$

$$
\begin{aligned}
&= \sigma^2 \alpha_{ab}(t_i)\alpha_{ab}(t_j)E\Big[\dot{W}(t_i)\dot{W}(t_j)\Big] - \sigma^2\alpha_{ab}(t_i)\int_a^b K_{DS}(t_j,\tau)E\Big[\dot{W}(t_i)\dot{W}(\tau)\Big]d\tau \\
&\quad - \sigma^2\alpha_{ab}(t_j)\int_a^b K_{DS}(t_i,s)E\Big[\dot{W}(t_j)\dot{W}(s)\Big]ds \\
&\quad + \sigma^2\int_a^b\int_a^b K_{DS}(t_i,s)K_{DS}(t_j,\tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau
\end{aligned} \tag{3.71}
$$

Applying the property (3.39) in the above equation.

$$
\begin{aligned}
&= \sigma^2 \alpha_{ab}(t_i)\alpha_{ab}(t_j)\delta(t_i - t_j) - \sigma^2 \alpha_{ab}(t_i) \int_a^b K_{DS}(t_j, \tau)\delta(t_i - \tau) d\tau \\
&\quad - \sigma^2 \alpha_{ab}(t_j) \int_a^b K_{DS}(t_i, s)\delta(t_j - s) ds + \sigma^2 \int_a^b K_{DS}(t_i, s) \int_a^b K_{DS}(t_j, \tau)\delta(s - \tau) d\tau ds
\end{aligned} \tag{3.72}
$$

Applying the property (3.44) in the above equation.

$$
\begin{aligned}
&= \sigma^2 \alpha_{ab}(t_i)\alpha_{ab}(t_j)\delta(t_i - t_j) - \sigma^2 \alpha_{ab}(t_i) K_{DS}(t_i, t_j) - \sigma^2 \alpha_{ab}(t_j) K_{DS}(t_j, t_i) \\
&\quad + \sigma^2 \int_a^b K_{DS}(t_i, s) K_{DS}(t_j, s) ds
\end{aligned} \tag{3.73}
$$

$$
\begin{aligned}
Cov[e_2(t_i), e_2(t_j)] &= E[e_2(t_i)e_2(t_j)] \\
&= \sigma^2 E\Big[\Big[\int_a^b \alpha_{ab}(s)\dot{W}(s)ds - \int_a^b K^1_{DS}(t_i,s)\dot{W}(s)ds\Big] \\
&\qquad \Big[\int_a^b \alpha_{ab}(\tau)\dot{W}(\tau)d\tau - \int_a^b K^1_{DS}(t_j,\tau)\dot{W}(\tau)d\tau\Big]\Big] \qquad (3.74)\\
&= \sigma^2 E\Big[\int_a^b\int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big] \\
&\qquad - \sigma^2 E\Big[\int_a^b\int_a^b \alpha_{ab}(s)\dot{W}(s)K^1_{DS}(t_j,\tau)\dot{W}(\tau)dsd\tau\Big] \\
&\qquad - \sigma^2 E\Big[\int_a^b\int_a^b \alpha_{ab}(\tau)\dot{W}(\tau)K^1_{DS}(t_i,s)\dot{W}(s)dsd\tau\Big] \\
&\qquad + \sigma^2 E\Big[\int_a^b\int_a^b K^1_{DS}(t_i,s)K^1_{DS}(t_j,\tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big] \qquad (3.75)\\
&= \sigma^2 \int_a^b\int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \\
&\qquad - \sigma^2 \int_a^b\int_a^b \alpha_{ab}(s)K^1_{DS}(t_j,\tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \\
&\qquad - \sigma^2 \int_a^b\int_a^b \alpha_{ab}(\tau)K^1_{DS}(t_i,s)E\Big[\dot{W}(\tau)\dot{W}(s)\Big]dsd\tau \\
&\qquad + \sigma^2 \int_a^b\int_a^b K^1_{DS}(t_i,s)K^1_{DS}(t_j,\tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \qquad (3.76)
\end{aligned}
$$

Applying the property (3.39) in the above equation.

$$
\begin{aligned}
&= \sigma^2\int_a^b\int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)\delta(s-\tau)dsd\tau - \sigma^2\int_a^b\int_a^b \alpha_{ab}(s)K^1_{DS}(t_j,\tau)\delta(s-\tau)dsd\tau \\
&\qquad - \sigma^2\int_a^b\int_a^b \alpha_{ab}(\tau)K^1_{DS}(t_i,s)\delta(s-\tau)dsd\tau \\
&\qquad + \sigma^2\int_a^b\int_a^b K^1_{DS}(t_i,s)K^1_{DS}(t_j,\tau)\delta(s-\tau)dsd\tau \qquad (3.77)
\end{aligned}
$$

Applying the property (3.44) in the above equation.

$$\begin{aligned}
&= \sigma^2 \int_a^b \alpha_{ab}(s)\alpha_{ab}(s)ds - \sigma^2 \int_a^b \alpha_{ab}(s)K_{DS}^1(t_j, s)ds \\
&\qquad - \sigma^2 \int_a^b \alpha_{ab}(s)K_{DS}^1(t_i, s)ds + \sigma^2 \int_a^b K_{DS}^1(t_i, s)K_{DS}^1(t_j, s)ds
\end{aligned} \tag{3.78}$$

$$\begin{aligned}
Cov[e_3(t_i), e_3(t_j)] &= E[e_3(t_i)e_3(t_j)] \\
&= \sigma^2 E\Big[\Big[\int_a^b \alpha_{ab}(s)(t_i - s)\dot{W}(s)ds - \int_a^b K_{DS}^2(t_i, s)\dot{W}(s)ds\Big] \\
&\qquad \Big[\int_a^b \alpha_{ab}(\tau)(t_j - \tau)\dot{W}(\tau)d\tau - \int_a^b K_{DS}^2(t_j, \tau)\dot{W}(\tau)d\tau\Big]\Big]
\end{aligned} \tag{3.79}$$

$$\begin{aligned}
&= \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)(t_i - s)(t_j - \tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big] \\
&\qquad - \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(s)(t_i - s)\dot{W}(s)K_{DS}^2(t_j, \tau)\dot{W}(\tau)dsd\tau\Big] \\
&\qquad - \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)\dot{W}(\tau)K_{DS}^2(t_i, s)\dot{W}(s)dsd\tau\Big] \\
&\qquad + \sigma^2 E\Big[\int_a^b \int_a^b K_{DS}^2(t_i, s)K_{DS}^2(t_j, \tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big]
\end{aligned} \tag{3.80}$$

$$\begin{aligned}
&= \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)(t_i - s)(t_j - \tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \\
&\qquad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)(t_i - s)K_{DS}^2(t_j, \tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \\
&\qquad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)K_{DS}^2(t_i, s)E\Big[\dot{W}(\tau)\dot{W}(s)\Big]dsd\tau \\
&\qquad + \sigma^2 \int_a^b \int_a^b K_{DS}^2(t_i, s)K_{DS}^2(t_j, \tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau
\end{aligned} \tag{3.81}$$

Applying the property (3.39) in the above equation.

$$
\begin{aligned}
&= \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)(t_i - s)(t_j - \tau)\delta(s-\tau)dsd\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)(t_i - s)K_{DS}^2(t_j,\tau)\delta(s-\tau)dsd\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)K_{DS}^2(t_i,s)\delta(s-\tau)dsd\tau \\
&\quad + \sigma^2 \int_a^b \int_a^b K_{DS}^2(t_i,s)K_{DS}^2(t_j,\tau)\delta(s-\tau)dsd\tau
\end{aligned} \tag{3.82}
$$

Applying the property (3.44) in the above equation.

$$
\begin{aligned}
&= \sigma^2 \int_a^b \alpha_{ab}(s)\alpha_{ab}(s)(t_i - s)(t_j - s)ds - \sigma^2 \int_a^b \alpha_{ab}(s)(t_i - s)K_{DS}^2(t_j,s)ds \\
&\quad - \sigma^2 \int_a^b \alpha_{ab}(s)(t_j - s)K_{DS}^2(t_i,s)ds + \sigma^2 \int_a^b K_{DS}^2(t_i,s)K_{DS}^2(t_j,s)ds
\end{aligned} \tag{3.83}
$$

$Cov[e_1(t_i), e_2(t_j)] = Cov[e_2(t_i), e_1(t_j)] = E[e_1(t_i)e_2(t_j)]$

$$
= \sigma^2 E\Bigg[\Big[\alpha_{ab}(t_i)\dot{W}(t_i) - \int_a^b K_{DS}^1(t_i,s)\dot{W}(s)ds\Big]\Big[\int_a^b \alpha_{ab}(\tau)\dot{W}(\tau)d\tau - \int_a^b K_{DS}^2(t_j,\tau)\dot{W}(\tau)d\tau\Big]\Bigg] \tag{3.84}
$$

$$
\begin{aligned}
&= \sigma^2 E\Big[\int_a^b \alpha_{ab}(t_i)\alpha_{ab}(\tau)\dot{W}(t_i)\dot{W}(\tau)d\tau\Big] - \sigma^2 E\Big[\int_a^b \alpha_{ab}(t_i)\dot{W}(t_i)K_{DS}^2(t_j,\tau)\dot{W}(\tau)d\tau\Big] \\
&\quad - \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(\tau)\dot{W}(\tau)K_{DS}^1(t_i,s)\dot{W}(s)ds\Big] \\
&\quad + \sigma^2 E\Big[\int_a^b \int_a^b K_{DS}^1(t_i,s)K_{DS}^2(t_j,\tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big]
\end{aligned} \tag{3.85}
$$

$$
\begin{aligned}
&= \sigma^2 \int_a^b \alpha_{ab}(t_i)\alpha_{ab}(\tau)E\Big[\dot{W}(t_i)\dot{W}(\tau)\Big]d\tau - \sigma^2 \int_a^b \alpha_{ab}(t_i)K_{DS}^2(t_j,\tau)E\Big[\dot{W}(t_i)\dot{W}(\tau)\Big]d\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)K_{DS}^1(t_i,s)E\Big[\dot{W}(\tau)\dot{W}(s)\Big]dsd\tau \\
&\quad + \sigma^2 \int_a^b \int_a^b K_{DS}^1(t_i,s)K_{DS}^2(t_j,\tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau
\end{aligned} \tag{3.86}
$$

Applying the property (3.39) in the above equation.

$$= \sigma^2 \int_a^b \alpha_{ab}(t_i)\alpha_{ab}(\tau)\delta(t_i-\tau)d\tau - \sigma^2 \int_a^b \alpha_{ab}(t_i)K^2_{DS}(t_j,\tau)\delta(t_i-\tau)d\tau$$
$$- \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)K^1_{DS}(t_i,s)\delta(s-\tau)dsd\tau$$
$$+ \sigma^2 \int_a^b \int_a^b K^1_{DS}(t_i,s)K^2_{DS}(t_j,\tau)\delta(s-\tau)dsd\tau \tag{3.87}$$

Applying the property (3.44) in the above equation.

$$= \sigma^2\alpha_{ab}(t_i)\alpha_{ab}(t_i) - \sigma^2\alpha_{ab}(t_i)K^2_{DS}(t_i,t_j) - \sigma^2 \int_a^b \alpha_{ab}(\tau)K^1_{DS}(t_i,\tau)d\tau$$
$$+ \sigma^2 \int_a^b K^1_{DS}(t_i,s)K^2_{DS}(t_j,s)ds \tag{3.88}$$

$$Cov[e_1(t_i),e_3(t_j)] = Cov[e_3(t_i),e_1(t_j)] = E[e_1(t_i)e_3(t_j)]$$
$$= \sigma^2 E\Big[\Big[\alpha_{ab}(t_i)\dot{W}(t_i) - \int_a^b K^1_{DS}(t_i,s)\dot{W}(s)ds\Big]\Big[\int_a^b \alpha_{ab}(\tau)(t_j-\tau)\dot{W}(\tau)d\tau$$
$$- \int_a^b K^3_{DS}(t_j,\tau)\dot{W}(\tau)d\tau\Big]\Big] \tag{3.89}$$
$$= \sigma^2 E\Big[\int_a^b \alpha_{ab}(t_i)\alpha_{ab}(\tau)(t_j-\tau)\dot{W}(t_i)\dot{W}(\tau)d\tau\Big] - \sigma^2 E\Big[\int_a^b \alpha_{ab}(t_i)\dot{W}(t_i)K^3_{DS}(t_j,\tau)\dot{W}(\tau)d\tau\Big]$$
$$- \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(\tau)(t_j-\tau)\dot{W}(\tau)K^1_{DS}(t_i,s)\dot{W}(s)ds\Big]$$
$$+ \sigma^2 E\Big[\int_a^b \int_a^b K^1_{DS}(t_i,s)K^3_{DS}(t_j,\tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big] \tag{3.90}$$
$$= \sigma^2 \int_a^b \alpha_{ab}(t_i)\alpha_{ab}(\tau)(t_j-\tau)E\Big[\dot{W}(t_i)\dot{W}(\tau)\Big]d\tau - \sigma^2 \int_a^b \alpha_{ab}(t_i)K^3_{DS}(t_j,\tau)E\Big[\dot{W}(t_i)\dot{W}(\tau)\Big]d\tau$$
$$- \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)(t_j-\tau)K^1_{DS}(t_i,s)E\Big[\dot{W}(\tau)\dot{W}(s)\Big]dsd\tau$$
$$+ \sigma^2 \int_a^b \int_a^b K^1_{DS}(t_i,s)K^3_{DS}(t_j,\tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \tag{3.91}$$

Applying the property (3.39) in the above equation.

$$
\begin{aligned}
&= \sigma^2 \int_a^b \alpha_{ab}(t_i)\alpha_{ab}(\tau)(t_j - \tau)\delta(t_i - \tau)d\tau - \sigma^2 \int_a^b \alpha_{ab}(t_i)K^3_{DS}(t_j, \tau)\delta(t_i - \tau)d\tau \\
&\qquad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)K^1_{DS}(t_i, s)\delta(s - \tau)dsd\tau \\
&\qquad + \sigma^2 \int_a^b \int_a^b K^1_{DS}(t_i, s)K^3_{DS}(t_j, \tau)\delta(s - \tau)dsd\tau
\end{aligned}
\tag{3.92}
$$

Applying the property (3.44) in the above equation.

$$
\begin{aligned}
&= \sigma^2\alpha_{ab}(t_i)\alpha_{ab}(t_i)(t_j - t_i) - \sigma^2\alpha_{ab}(t_i)K^3_{DS}(t_i, t_j) \\
&\qquad - \sigma^2 \int_a^b \alpha_{ab}(s)(t_j - s)K^1_{DS}(t_i, s)ds + \sigma^2 \int_a^b K^1_{DS}(t_i, s)K^3_{DS}(t_j, s)ds
\end{aligned}
\tag{3.93}
$$

$$
\begin{aligned}
Cov[e_2(t_i), e_3(t_j)] = Cov[e_3(t_i), e_2(t_j)] &= E[e_2(t_i)e_3(t_j)] \\
&= \sigma^2 E\Big[\Big[\int_a^b \alpha_{ab}(s)\dot{W}(s)ds - \int_a^b K_{DS}^2(t_i, s)\dot{W}(s)ds\Big] \\
&\quad \Big[\int_a^b \alpha_{ab}(\tau)(t_j - \tau)\dot{W}(\tau)d\tau - \int_a^b K_{DS}^3(t_j, \tau)\dot{W}(\tau)d\tau\Big]\Big] \qquad (3.94) \\
&= \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)(t_j - \tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big] \\
&\quad - \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(s)\dot{W}(s)K_{DS}^3(t_j, \tau)\dot{W}(\tau)dsd\tau\Big] \\
&\quad - \sigma^2 E\Big[\int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)\dot{W}(\tau)K_{DS}^2(t_i, s)\dot{W}(s)dsd\tau\Big] \\
&\quad + \sigma^2 E\Big[\int_a^b \int_a^b K_{DS}^2(t_i, s)K_{DS}^3(t_j, \tau)\dot{W}(s)\dot{W}(\tau)dsd\tau\Big] \qquad (3.95) \\
&= \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)(t_j - \tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)K_{DS}^3(t_j, \tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)K_{DS}^2(t_i, s)E\Big[\dot{W}(\tau)\dot{W}(s)\Big]dsd\tau \\
&\quad + \sigma^2 \int_a^b \int_a^b K_{DS}^2(t_i, s)K_{DS}^3(t_j, \tau)E\Big[\dot{W}(s)\dot{W}(\tau)\Big]dsd\tau \qquad (3.96)
\end{aligned}
$$

Applying the property (3.39) in the above equation.

$$
\begin{aligned}
&= \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)\alpha_{ab}(\tau)(t_j - \tau)\delta(s - \tau)dsd\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(s)K_{DS}^3(t_j, \tau)\delta(s - \tau)dsd\tau \\
&\quad - \sigma^2 \int_a^b \int_a^b \alpha_{ab}(\tau)(t_j - \tau)K_{DS}^2(t_i, s)\delta(s - \tau)dsd\tau \\
&\quad + \sigma^2 \int_a^b \int_a^b K_{DS}^2(t_i, s)K_{DS}^3(t_j, \tau)\delta(s - \tau)dsd\tau \qquad (3.97)
\end{aligned}
$$

Applying the property (3.44) in the above equation.

$$= \sigma^2 \int_a^b \alpha_{ab}(s)\alpha_{ab}(s)(t_j - s)ds - \sigma^2 \int_a^b \alpha_{ab}(s)K_{DS}^3(t_j, s)ds$$
$$- \sigma^2 \int_a^b \alpha_{ab}(s)(t_j - s)K_{DS}^2(t_i, s)ds + \sigma^2 \int_a^b K_{DS}^2(t_i, s)K_{DS}^3(t_j, s)ds \qquad (3.98)$$

Covariance matrix=

$$\begin{bmatrix} Cov[e_1(t_i), e_1(t_j)] & Cov[e_1(t_i), e_2(t_j)] & Cov[e_1(t_i), e_3(t_j)] \\ Cov[e_2(t_i), e_1(t_j)] & Cov[e_2(t_i), e_2(t_j)] & Cov[e_2(t_i), e_3(t_j)] \\ Cov[e_3(t_i), e_1(t_j)] & Cov[e_3(t_i), e_2(t_j)] & Cov[e_3(t_i), e_3(t_j)] \end{bmatrix} \qquad (3.99)$$

The elements of the covariance matrix are not constant; hence the error terms are heteroskedastic. The covariance matrix is updated in every step in the recursive GLS estimator.

3.3.2 Error covariance matrix for n^{th} order system [5]

When we calculate the error covariance matrix for an n^{th} order system, (3.99) changes to:

$$[S_n]^{-1} := \begin{bmatrix} \text{Cov}[e^n(t_1), e^n(t_1)] \cdots \text{Cov}[e^n(t_1), e^n(t_N)] \\ \ddots \\ \text{Cov}[e^n(t_N), e^n(t_1)] \cdots \text{Cov}[e^n(t_N), e^n(t_N)] \end{bmatrix}; \quad n = 1, 2, 3, ...$$

where,

$$\begin{aligned}
&\text{Cov}[e^n(t_i), e^n(t_j)] = E[e^n(t_i)e^n(t_j)] \\
&= \frac{\sigma^2}{(n-1)!}^2 \int_a^b \alpha_{ab}(s)\alpha_{ab}(s)(t_i - s)^{n-1}(t_j - s)^{n-1}ds \\
&\qquad - \frac{\sigma^2}{(n-1)!}\int_a^b \alpha_{ab}(s)(t_i - s)^{n-1}K_{DS_n,y}(t_j, s)ds \\
&\qquad - \frac{\sigma^2}{(n-1)!}\int_a^b \alpha_{ab}(s)(t_j - s)^{n-1}K_{DS_n,y}(t_i, s)ds + \sigma^2\int_a^b K_{DS_n,y}(t_i, s)K_{DS_n,y}(t_j, s)ds
\end{aligned} \tag{3.100}$$

where the matrices $S_n \in \mathbb{R}^{N\times N}$ defined as $S := \text{diag}(S_1, ..., S_n)$ for $n = 1, 2, 3$ and K_{DS_n} is a kernel of n^{th} order as defined in (2.8).

3.4 Recursive GLS algorithm [6]

The covariance matrix depends on the unknown variance σ^2, and the unknown parameter vector $\bar{a}$ in the K_{DS} kernels. Hence the standard GLS cannot be applied directly. This paves the way to employ the modified recursive GLS [6] [19] in which the covariance matrix is estimated progressively. Letting $Q_i - P_i\bar{a}$ denote the regression error e_i in batch i, the modified recursive GLS algorithm computes

$$\hat{a}_k = \arg\min_{\bar{a}} \left(\sum_{i=1}^{k} (Q_i - P_i\bar{a})^T S_i (Q_i - P_i\bar{a}) \right) \tag{3.101}$$

where $\hat{a}_k$ is the parameter estimate update at iteration k of the algorithm. Each weighting matrix S_{k+1}, is calculated as the inverse of the covariance matrix based on the parameter and variance estimates i.e. $\hat{a}_k$ and σ^2, obtained from the residual trajectory $y_M(t) - y_E(t)$ in previous iteration k, where y_E signifies the estimated output. A modified recursive GLS algorithm with inverse covariance weighting is explained below

At iteration $k+1$, the algorithm strives to minimize

$$\begin{aligned} &\min(\bar{e}_{k+1}^T \bar{S}_{k+1} \bar{e}_{k+1}) \\ \text{subject to:} \quad &\bar{Q}_{k+1} = \bar{P}_{k+1}\bar{a}_{k+1} + \bar{e}_{k+1} \end{aligned} \tag{3.102}$$

where

$$\bar{Q}_{k+1} = \begin{bmatrix} Q_0 \\ Q_1 \\ \vdots \\ Q_{k+1} \end{bmatrix}; \quad \bar{P}_{k+1} = \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_{k+1} \end{bmatrix}; \quad \bar{e} = \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{k+1} \end{bmatrix}$$

and

$$\bar{S}_{k+1} = \operatorname{diag}(S_0, S_1, \ldots, S_{k+1}) \tag{3.103}$$

The solution of the above is written as

$$(\bar{P}_{k+1}^T \bar{S}_{k+1} \bar{P}_{k+1})\hat{a}_{k+1} = \bar{P}_{k+1}^T \bar{S}_{k+1} \bar{Q}_{k+1} \tag{3.104}$$

or in summation form as

$$\left(\sum_{i=0}^{k+1} P_i^T S_i P_i \right) \hat{a}_{k+1} = \sum_{i=0}^{k+1} P_i^T S_i Q_i \tag{3.105}$$

Defining

$$M_{k+1} = \sum_{i=0}^{k+1} P_i^T S_i P_i \tag{3.106}$$

the recursion for M_{k+1} is:

$$M_{k+1} = M_k + P_{k+1}^T S_{k+1} P_{k+1} \tag{3.107}$$

Rearranging (3.105) gives

$$\begin{aligned}\hat{a}_{k+1} &= M_{k+1}^{-1}\left[\left(\sum_{i=0}^{k} P_i^T S_i P_i\right)\hat{a}_k + P_{k+1}^T S_{k+1} Q_{k+1}\right] \\ &= M_{k+1}^{-1}\left[M_k \hat{a}_k + P_{k+1}^T S_{k+1} Q_{k+1}\right] \qquad (3.108)\end{aligned}$$

Another form of (3.108) is delivered by the recursion (3.107) and reads

$$\begin{aligned}\hat{a}_{k+1} &= \hat{a}_k - M_{k+1}^{-1}(P_{k+1}^T S_{k+1} P_{k+1}\hat{a}_k - P_{k+1}^T S_{k+1} Q_{k+1}) \\ &= \hat{a}_k + M_{k+1}^{-1} P_{k+1}^T S_{k+1}(Q_{k+1} - P_{k+1}\hat{a}_k) \qquad (3.109)\end{aligned}$$

A recursion for M_{k+1}^{-1} is obtained by applying the following identity to the recursion in (3.107)

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(DA^{-1}B + C^{-1})^{-1}DA^{-1} \qquad (3.110)$$

which yields

$$M_{k+1}^{-1} = M_k^{-1} - M_k^{-1}P_{k+1}^T(P_{k+1}M_k^{-1}P_{k+1}^T + S_{k+1}^{-1})^{-1}P_{k+1}M_k^{-1}$$

Defining $R_{k+1} = M_{k+1}^{-1}$ the latter becomes

$$R_{k+1} = R_k - R_k P_{k+1}^T(S_{k+1}^{-1} + P_{k+1}R_k P_{k+1}^T)^{-1}P_{k+1}R_k \qquad (3.111)$$

Equations (3.110) and (3.111) constitute the recursive GLS least squares algorithm.

3.5 Algorithmic implementation of the RLS

The pseudo code for the RLS algorithm based on the above given equations is given as follows:

Note: before running the following algorithm ensure you have:

- y: The signal to be estimated.

- t: The time axis of y.
- knots: The number of knots at which the RLS algorithm would calculate the forward and backward kernels.
- mode: the sampling strategy with respect to the location of points to be sampled. example: mid-point, end-point, start of the trajectory and so on.
- max integrations: The maximum number of integrations to be performed in order to obtain the most integrated regression term for multiple regression.
- tolerance: a metric to decide the accuracy of parameters and check if further recursion is needed

Algorithm 1 Calculate a_0, a_1, a_2

```
$t_{knots} \leftarrow$ get knots
$a_k \leftarrow$ initialize with [0.0, 0.0, 0.0]
**while** diff $<$ tolerance **do**
    $t_k \leftarrow$ get single knot
    P, Q $\leftarrow$ obtain P and Q matrix
    $y_{estimate} \leftarrow$ obtain by Gramm-Schmidt orthonormalization
    $var \leftarrow$ variance of y -$y_{estimate}$
    $S_{k+1} \leftarrow$ Calculate S
    $R_{k+1} \leftarrow$ Calculate $R_{k+1}$
    $a_{k+1} \leftarrow$ Calculate $a_{k+1}$
    $diff \leftarrow$ norm(ak- $a_{k+1}$)
**end while**
**return** $a_k$
```

3.6 Reconstruction of the output derivatives [5]

Given a measurement process realization $\overline{y_M}$ on $[a, b]$, the derivatives can be reconstructed using,

$$y^{(i)}(t) = \int_a^b K_{DS}^i(t,\tau)\hat{y}(\tau)\mathrm{d}\tau \tag{3.112}$$

where, K^i_{DS} are the kernel representation for output derivatives. Once the parameters of the system are estimated successfully, the system output can be reconstructed by projection onto the finite dimensional subspace of the RKHS spanned by the fundamental solutions of the characteristic equation. This is because every solution of the characteristic equation with the already identified parameter vector $\bar{a}$ satisfies the reproducing property so, the projection onto the space of fundamental solutions will be the noise free trajectory of the system. The fundamental solutions are found by integrating the characteristic equation for n sets of initial conditions.

These initial conditions can be taken as canonical basis vectors in $\mathbb{R}^n$ i.e,

$$\begin{aligned} e_1 &= [1, 0, ..., 0] \\ e_2 &= [0, 1, ..., 0] \\ &\vdots \\ e_n &= [0, 0, ..., 1] \end{aligned} \tag{3.113}$$

The system equation (3.25) is then solved for each individual initial condition yielding solutions

$$\overline{y_k}(\bar{a}) = e_k \qquad k = 1, \cdots, n$$

It is an elementary fact from the theory of ordinary differential equations that any solution of the system (3.25) with any initial condition is a linear combination of such fundamental solutions $\overline{y_i}$. Hence we search for the coefficients of this linear combination so that the resulting function is the closest to the output measurement data. Closest solution is found in terms of an orthogonal projection onto span of S^a.

$$S^a = \text{span}\,\{\overline{y}_k(\cdot), k = 1, \cdots, n\}$$

This is best done by orthonormalizing the set of fundamental solutions.The projection of a measured noisy signal $y_M(\cdot) \in L^2[a, b]$ into S^a is given as,

$$y_E(\cdot) \triangleq \arg\min \left\{ \|y_M - y\|_2^2 ,\ y \in S^a \right\} \tag{3.114}$$

We seek,

$$y_E = \sum_{k=1}^{n} \hat{c}_k \overline{y}_k \tag{3.115}$$

As y_E is a linear combination of orthonormalized vectors, the optimality conditions in (3.114) is achieved if and only if,

$$\left\langle y_M | \overline{y}_j \right\rangle_2 = \sum_{k=1}^{n} \hat{c}_i \left\langle \overline{y}_k | \overline{y}_j \right\rangle_2 \;\; k = 1, \cdots, n \tag{3.116}$$

which can be written in a matrix form as:

$$\begin{aligned} &v = G(\overline{y})\hat{c}; \quad G(\overline{y}) \triangleq \text{mat}\left\{\left\langle \overline{y}_k | \overline{y}_j \right\rangle_2\right\}_{k,j=1}^{n} \\ &v \triangleq \text{vec}\left\{\langle y_M | \overline{y}_k \rangle_2\right\}_{k=1}^{n}; \;\; \hat{c} \triangleq \text{vec}\left\{\hat{c}_k\right\}_{k=1}^{n} \end{aligned} \tag{3.117}$$

G is called the Gram matrix for vectors in span S^a and is invertible because it is known that all fundamental solutions are linearly independent, from the theory of differential equations.

$$\hat{c} = G^{-1}(\overline{y})v \tag{3.118}$$

y_E, the estimated output is thus obtained from (3.115). The estimates of the derivatives $y_E^{(i)}$, $i = 1, \cdots, n$ are obtained by way of integral transforms of y_E.

3.7 Results and Discussion

We employed the kernel based multiple regression algorithm for parameter and state estimation as defined earlier in this chapter .We compare our results with Ghoshal et al [33] and John et al [34] for linear time invariant systems to start off and then further results are presented in the same order as the chapters are presented throughout this thesis
The following key metrics will be used to compare the performance of our algorithms with the true system response:

- Root Mean Square Deviation (RMSD):

$$RMSD = \sqrt{\sum_{i=1}^{N} \frac{1}{n} [y_T(t_i) - y_E(t_i)]^2}; \tag{3.119}$$

- Maximum Absolute Deviation (MAD):

$$MAD = \max_{t \in [a,b]} |y_T(t) - y_E(t)| \tag{3.120}$$

3.7.1 Recursive Least Squares Applied to LTI system:

Example 1: Let us consider a third order LTI system as mentioned below [26].

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -26 & -13 & 0 \end{bmatrix} x \; ; y = x_1 \; ; \; x(0) = [0, 0, 1] \tag{3.121}$$

with its corresponding characteristic equation

$$y^{(3)}(t) + a_2 y^{(2)}(t) + a_1 y^{(1)} + a_0 y(t) = 0 \tag{3.122}$$

The parameters a_0, a_1, a_2 are assumed to be unknown (when in reality the values are 26, 13, 0 respectively).
The measured realization of the output y_M is obtained by adding Gaussian white noise to the nominal trajectory.

$$y_M = y_T + \epsilon \tag{3.123}$$

where y_T is the true trajectory and ϵ is a white noise, with

$$\epsilon \sim \mathcal{N}(\mu, \, \sigma^2) \tag{3.124}$$

We apply Gaussian white noise of zero mean ($\mu = 0$) and variance ranging from $\sigma = 1$ to $\sigma = 6$ at increment of 3 to the solution of the above system and the output curves are given below. Given here are the reference trajectories which we may call "ground truth":

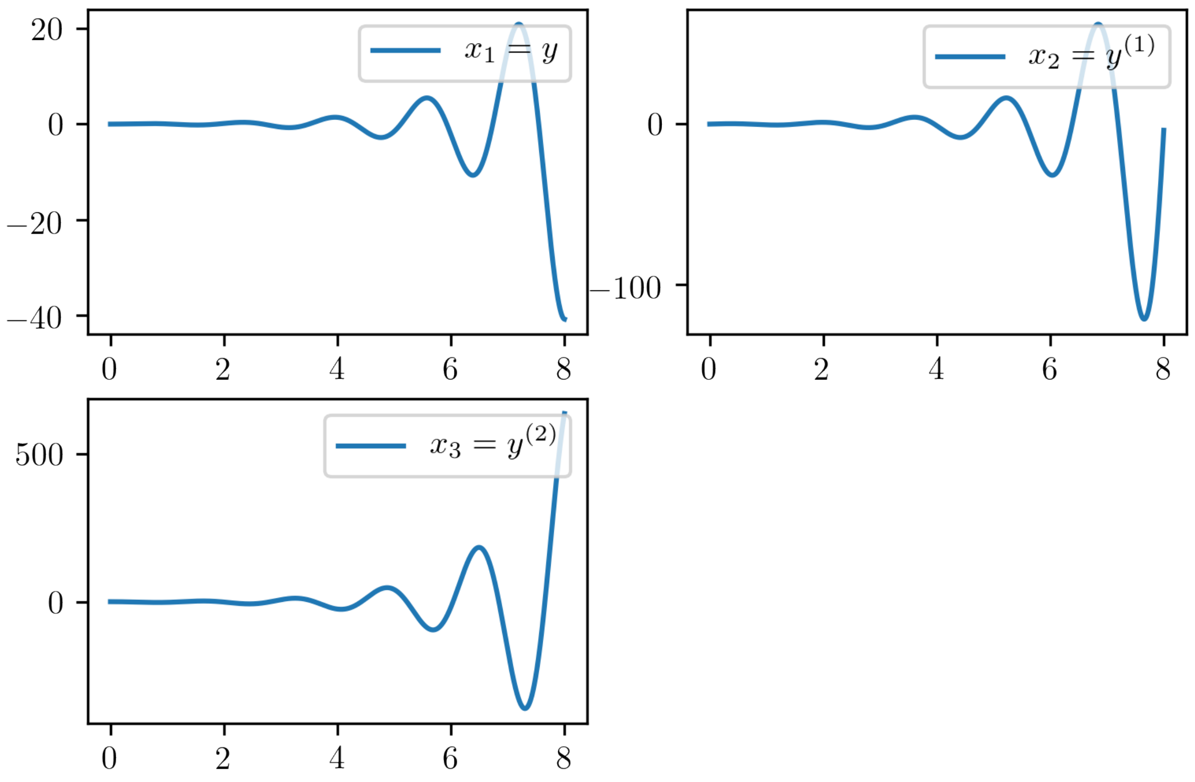

Figure 3.1 3^{rd} Order system ground truth

The recursive kernel based FGLS is employed in batches of N=2000 knots sampled randomly from a uniform distribution of 60,000 samples over [a,b] = [0,8]. The threshold value for stopping the algorithm is ϵ=0.01. White noise of $\sigma = 1$ (SNR = -5.84dB) is added to the system is subsequently increased and all the results are represented below in increasing order of noise variance:

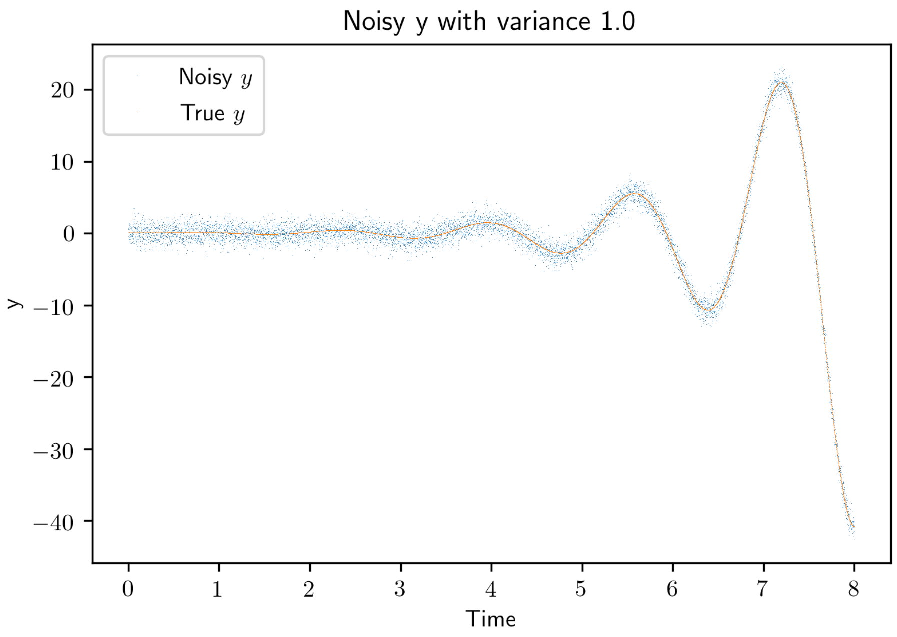

Figure 3.2 True and noisy output trajectories of the system with AWGN of $\mu = 0$ and $\sigma = 1$

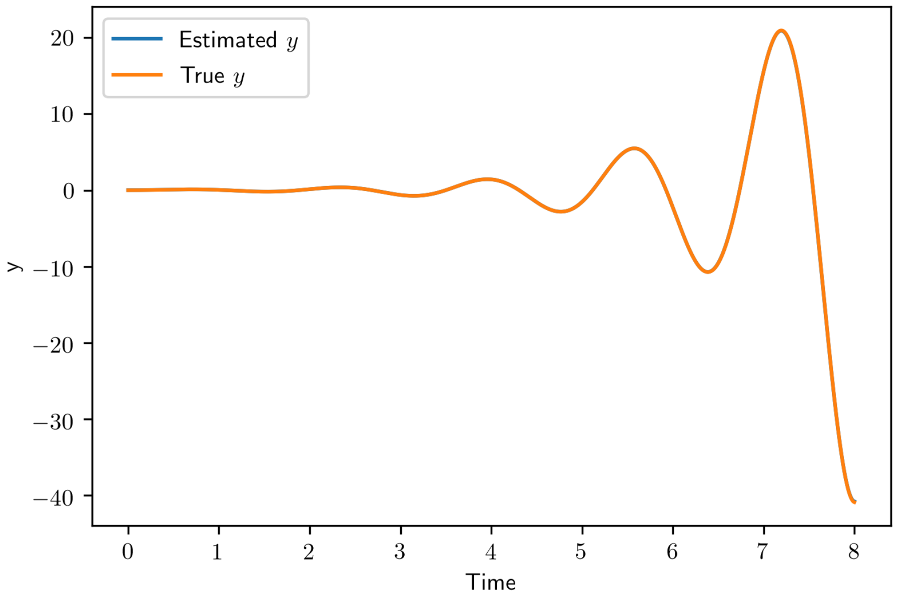

Figure 3.3 True and reconstructed signal of the system with AWGN of $\mu = 0$ and $\sigma = 1$

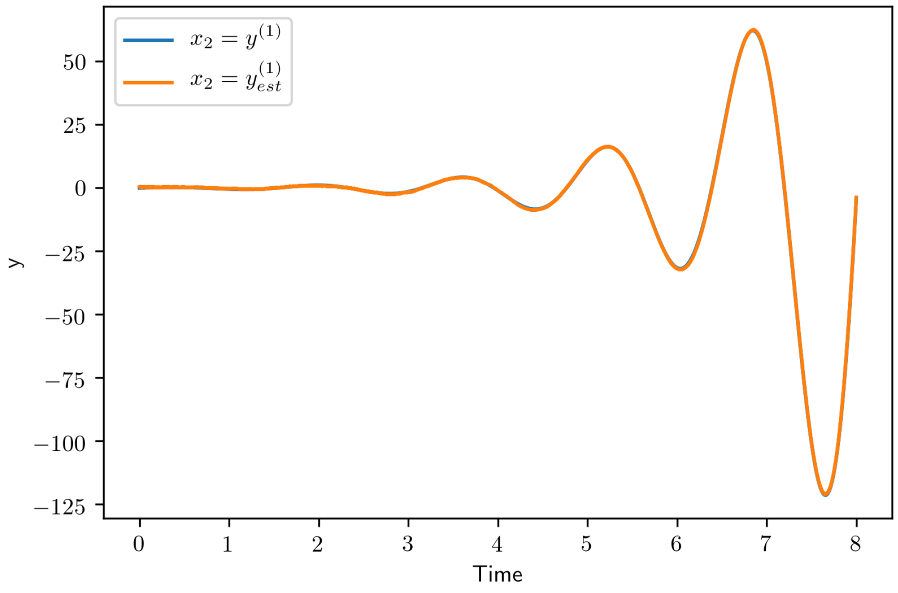

Figure 3.4 True and reconstructed first derivative of the system with AWGN of $\mu = 0$ and $\sigma = 1$

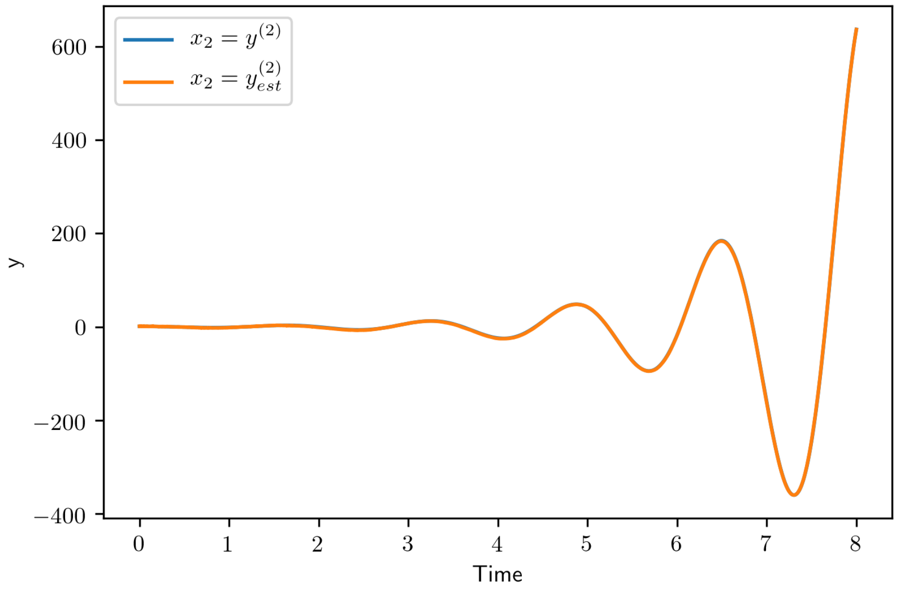

Figure 3.5 True and reconstructed Second derivative of the system with AWGN of $\mu = 0$ and $\sigma = 1$

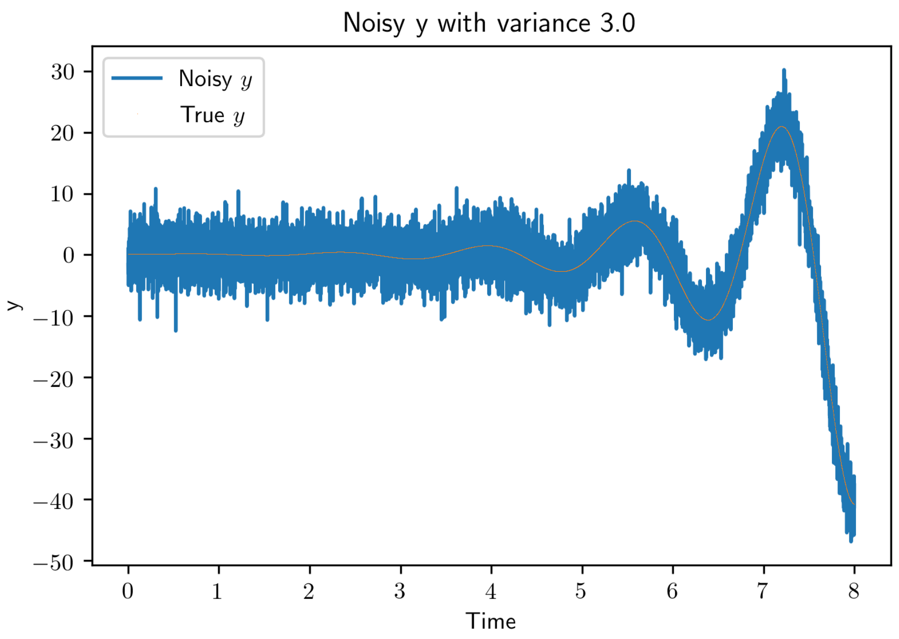

Figure 3.6 True and noisy output trajectories of the system with AWGN of $\mu = 0$ and $\sigma = 3$

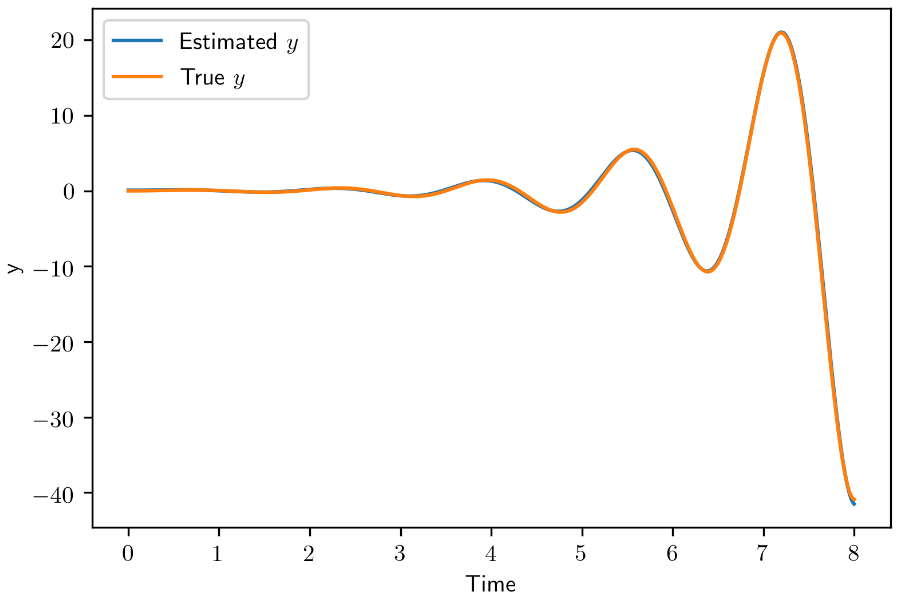

Figure 3.7 True and reconstructed signal of the system with AWGN of $\mu = 0$ and $\sigma = 3$

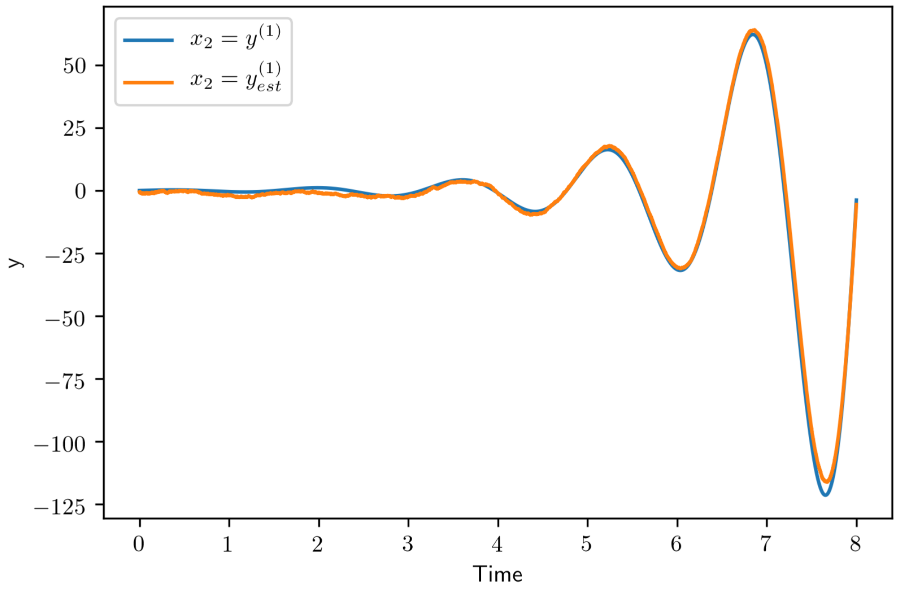

Figure 3.8 True and reconstructed first derivative of the system with AWGN of $\mu = 0$ and $\sigma = 3$

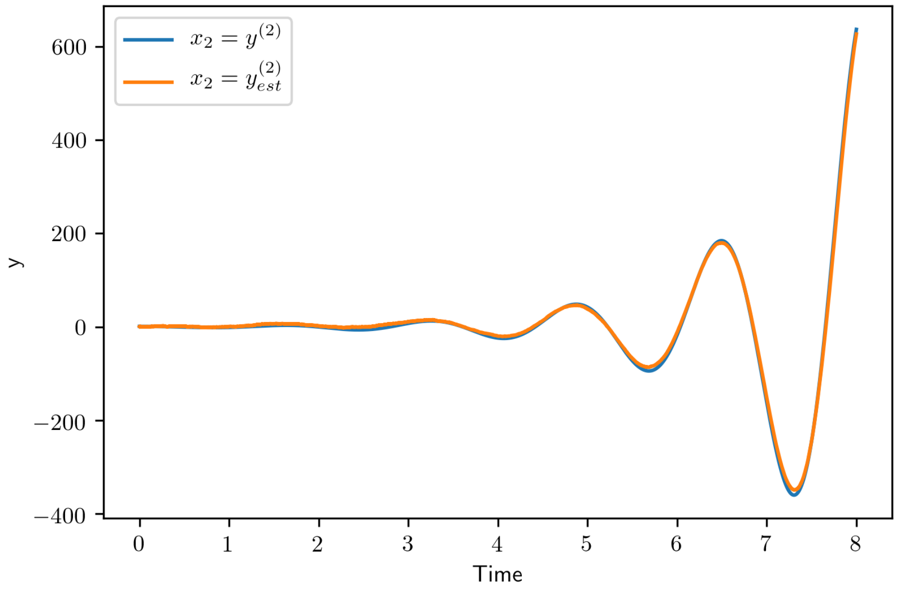

Figure 3.9 True and reconstructed Second derivative of the system with AWGN of $\mu = 0$ and $\sigma = 3$

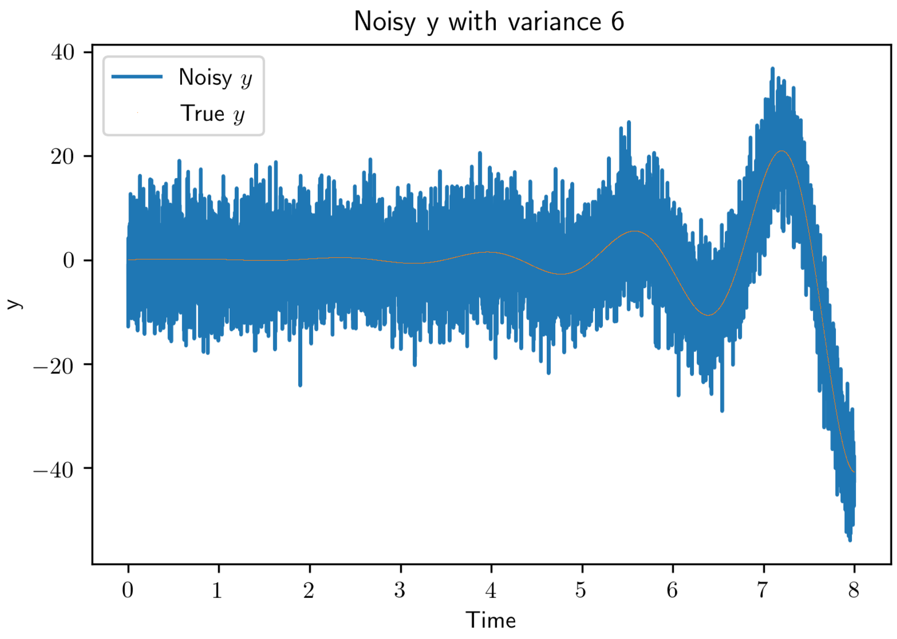

Figure 3.10 True and noisy output trajectories of the system with AWGN of $\mu = 0$ and $\sigma = 6$

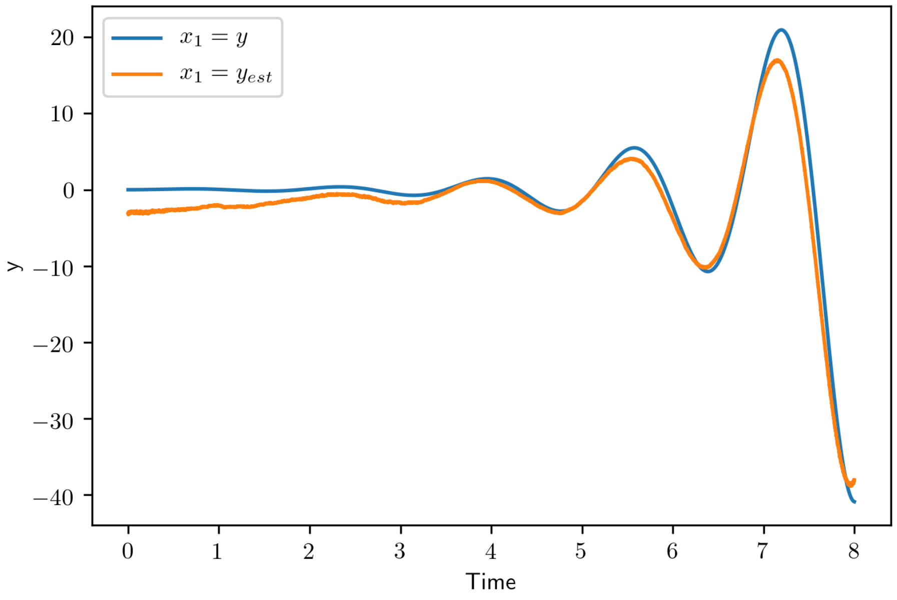

Figure 3.11 True and reconstructed signal of the system with AWGN of $\mu = 0$ and $\sigma = 6$

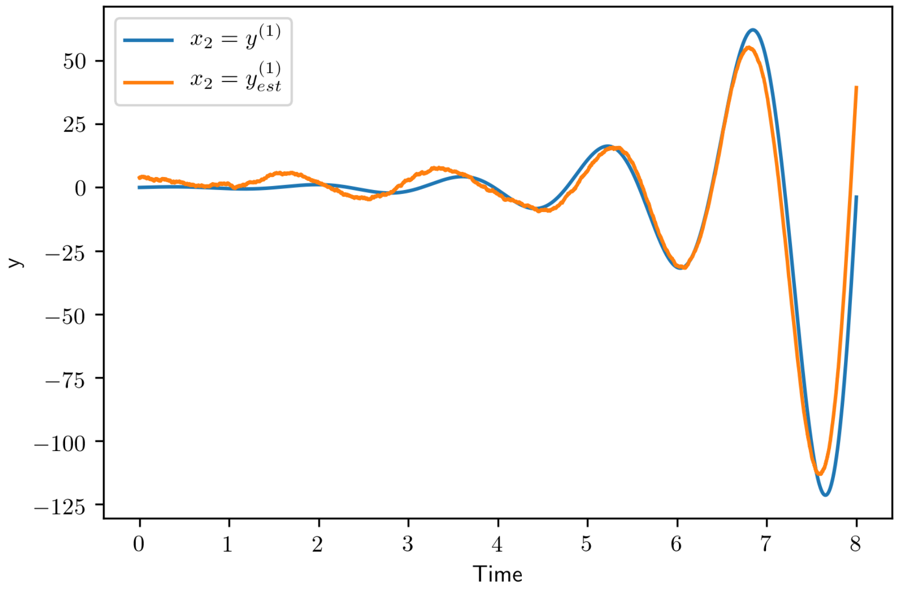

Figure 3.12 True and reconstructed first derivative of the system with AWGN of $\mu = 0$ and $\sigma = 6$

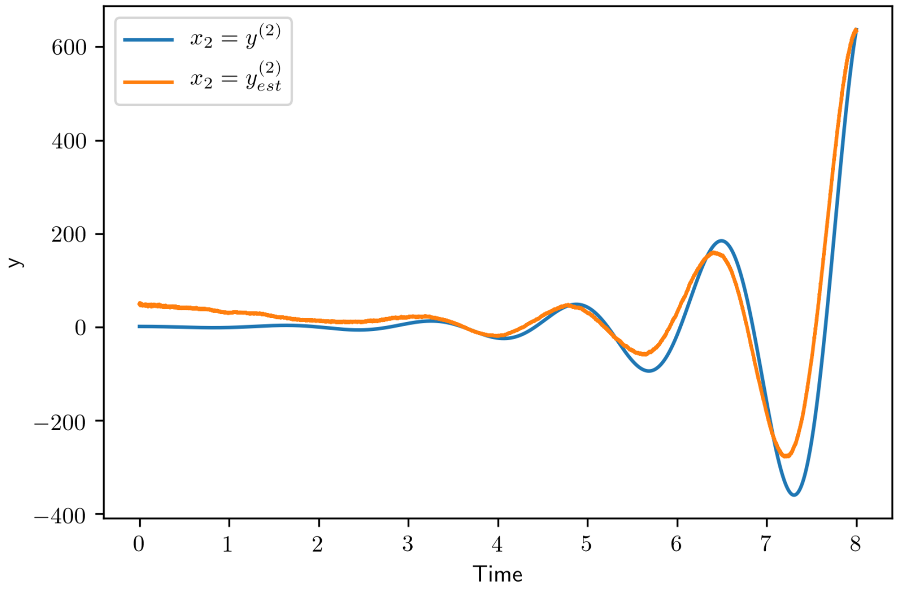

Figure 3.13 True and reconstructed Second derivative of the system with AWGN of $\mu = 0$ and $\sigma = 6$

The table given below represents all the results we obtain for increasing for consecutively increasing the noise and varying the number of knots N:

Noise	SNR(dB)	N	Estimated a	$RMSD$
$\sigma = 1$	4.2794	2000	26.0035, 12.9961, 0.0006	0.000284
		3000	26.0029, 13.0049, -0.0001	0.000214
		4000	26.0015, 13.0033, -0.0002	0.000196
$\sigma = 1.5$	0.8162	2000	26.0167, 12.9624 , -0.0002	0.000316
		3000	26.0091, 12.9822 , -0.0001	0.000291
		4000	26.0015, 12.9845 , -0.0003	0.000255
$\sigma = 2$	-1.6735	2000	26.0344, 12.5089, -0.0023	0.000725
		3000	26.0242, 12.4009, -0.0021	0.000671
		4000	26.0233, 12.221, -0.0008	0.000445
$\sigma = 3$	-5.1032	2000	25.5697, 12.6212, -0.0002	0.00207
		3000	25.6726, 12.8201, 0.0001	0.00157
		4000	25.6654, 12.7546, -0.0002	0.00260
$\sigma = 4$	-7.6043	2000	23.6565, 13.9919, -0.0631	0.00316
		3000	27.0470, 12.0477, -0.0711	0.00289
		4000	27.0226, 12.0321, -0.0001	0.00011
$\sigma = 6$	-11.1499	2000	16.7977, 14.7731, 0.1528	0.0227
		3000	17.1820, 14.7037, 0.1589	0.0215
		4000	18.1265, 14.2354, 0.1326	0.0190

Table 3.1 Estimates of parameter values and $RMSD$ for various SNR levels and sample size N

Table (3.2) shows the comparison of the proposed method with Ghoshal et al. [3] and John et al. [34] results. The number samples taken is 1000 from the uniform distribution in [0,8]. The example 1 in Section 5.1 is considered here.

Noise	*Method*	Estimated a_0	Estimated a_1	Estimated a_2	RMSD
$\sigma = 1$	FGLS	26.0035	12.9961	0.0006	0.0002
	Ghoshal et al. [3]	25.6977	13.2109	0.0020	0.0006
	John et al. [34]	34.2321	6.3355	-0.7380	0.0419
$\sigma = 3$	FGLS	26.0167	12.9624	-0.0002	0.0002
	Ghoshal et al. [3]	23.4432	13.6524	0.1397	0.0095
	John et al. [34]	7.3532	4.1899	-1.0202	0.1432
$\sigma = 6$	FGLS	26.0344	12.5089	-0.0023	0.0007
	Ghoshal et al. [3]	16.3091	8.5396	1.4127	0.0152
	John et al. [34]	1.0298	-8.0940	5.1837	0.3024

Table 3.2 Comparitive Study with Ghoshal et al. and John et al.

Chapter 4

Moving-horizon Estimation of Non-linear systems

So far in the previous chapters, we have dealt with purely linear time invariant systems. This chapter (and following) will deal only with non-linear systems. While, the techniques that were developed so far are also linear in their nature, we will adapt them in order to estimate non-linear systems. Starting off with applying FGLS directly to Non-linear systems, investigating the hyper-parameters and their influence on estimation accuracy. The shortcomings of FGLS on Non-linear systems are discussed and then we introduce the Moving-horizon estimator and analyze it's performance.

4.1 Bias-Variance trade-off [7]

We start with the classical setting of parametric estimation. Our goal is to construct an estimator for the unknown parameters θ^* given the observed set of noisy data $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$, where n is the size of the dataset.

In case of Kernel based linear regression, we denote our estimator as $\hat{\theta_n}$, where $\hat{\theta_n} = (X^T X)^{-1} X^T \vec{y}$. In the above model, $\hat{\theta_n}$ can be treated as a random variable even though the original system parameters θ^* may not be because $\hat{\theta_n}$ is a function of the noisy dataset S. Although the underlying behaviour of said randomness in $\hat{\theta_n}$ is still governed by θ^*. Mathematically, the variance and bias are the second and first moment respectively of $\hat{\theta_n}$'s sampling distribution. Unlike the bias term, the variance is not directly dependant on the

true parameters of the system.
The Bias and Variance of an estimator are not necessarily directly related (just as how the first and second moment of any distribution are not necessarily related). It is possible to have estimators that have high or low bias and have either high or low variance. Under the squared error, the Bias and Variance of an estimator are related as:

$$\begin{aligned}
\text{MSE}(\hat{\theta_n}) &= \mathbb{E}\left[\|\hat{\theta_n} - \theta^*\|^2\right] && (4.1)\\
&= \mathbb{E}\left[\|\hat{\theta_n} - \mathbb{E}[\hat{\theta_n}] + \mathbb{E}[\hat{\theta_n}] - \theta^*\|^2\right] && (4.2)\\
&= \mathbb{E}\left[\|\hat{\theta_n} - \mathbb{E}[\hat{\theta_n}]\|^2\right] + \|\,\mathbb{E}[\hat{\theta_n}] - \theta^*\|^2 && (4.3)\\
&= \mathbb{E}\left[tr\left[(\hat{\theta_n} - \mathbb{E}[\hat{\theta_n}])(\hat{\theta_n} - \mathbb{E}[\hat{\theta_n}])^T\right]\right] + \|\,\mathbb{E}[\hat{\theta_n}] - \theta^*\|^2 && (4.4)\\
&= tr\left[Var(\hat{\theta_n})\right] + \|Bias(\hat{\theta_n})\|^2 && (4.5)
\end{aligned}$$

Generally, When one tries to suppress the variance term through techniques such as regularization, the estimator starts getting biased and thereby ends up under-fitting the data. On the other hand, when we over-parameterize the estimator, we have a high degree of freedom to fit higher order curves but that ends up causing issues in the non-linear case as will be discussed throughout this chapter. Since, we don't sample everywhere on the curve, instead, only at certain points like mid-point, end-point or any other sampling point as we strategize, knot selection also plays a key.

4.2 Influence of model order on Non-linear estimation-

Consider the following linear state-space model:

$$\begin{aligned}
x_k &= Ax_{k-1} + Bu_{k-1} + w_{k-1} && (4.6)\\
z_k &= Hx_k + v_k && (4.7)
\end{aligned}$$

where $x \in \mathbb{R}^n, z \in \mathbb{R}^m, u \in \mathbb{R}^l$, $p(w) \sim N(0, Q)$ and $p(v) \sim N(0, R)$.

The problem mentioned earlier of the trade -off between a bias and a variance gets bigger in the non-linear case since we are using linear kernels to fit a not linear curve. The parameters that we identify at a specific point on the non-linear noisy curve, may not represent the whole curve, thereby introducing more inaccuracies in our estimated curves. As mentioned earlier, selection of a model order is key towards the estimation problem. Also, once the model order is fixed, an order needs to be specified before its parameters are estimated which, in our case can be treated as prior knowledge of the system. So even though we might not know the system dynamics, we would still need an idea as to how dynamic the system curves might be and therefore be able to predict an order. In general, our aim is to not use a model order which is higher than what is required or go lower to the point where we oversimplify our estimated model.

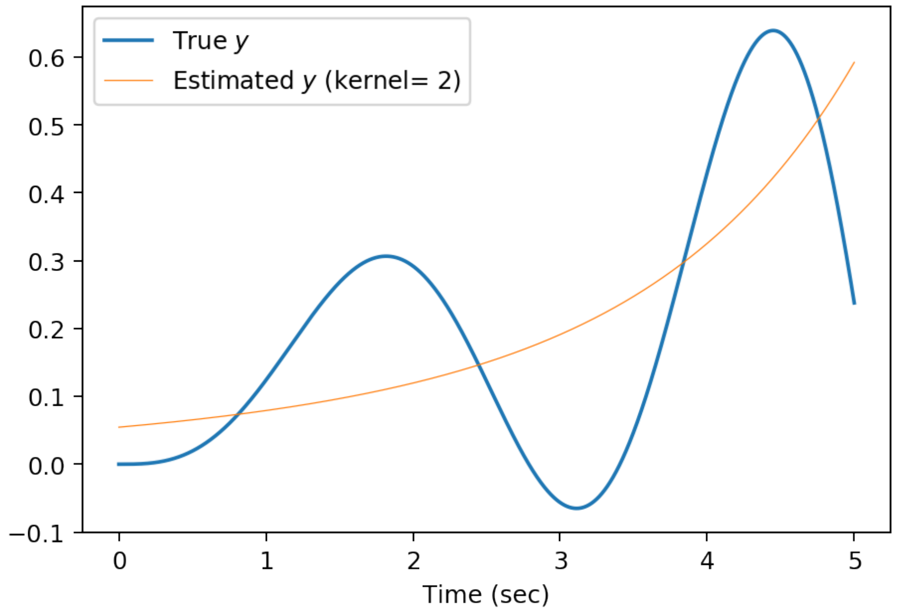

Figure 4.1 True y vs under-estimated y of order 2

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ a_0 * x_1^3 + a_1 * x_2 + a_2 * x_3 \end{bmatrix} \tag{4.8}$$

$$y = x_1 \tag{4.9}$$

$$x_{initial} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.10}$$

Given above is an example of 3^{rd} order non-linear curve that has been estimated by a linear curve of kernel order $n = 2$. The strategy employed in the above estimation case was to sample the noisy signal, calculate the double-sided kernels at the mid point and then estimate the rest of the curve based on those parameters. The linear system order(order=2) is lesser than the actual curve that is to be estimated($n = 4$), given that the signal spans over $t = 5$ seconds and has $N = 20,000$ samples from the noisy signal, it is evident that it only fits at the absolute center of the curve while completely missing the rest of the curve. The above given curve is an edge case with a particularly bad estimation but which explains the concept of order selection quite accurately.
The above problem can be solved by increasing the number of points at which the signals are sampled and subsequently where the kernels are calculated and parameters estimated. The problem of under-fitting a curve can also be solved by increasing the model order. However, it remains a key trade off not to select an order too high as to over-fit on just one curve which may or may not generalize well. There exist more informed ways of model selection such as the Bayesian information criterion or the general information criterion.

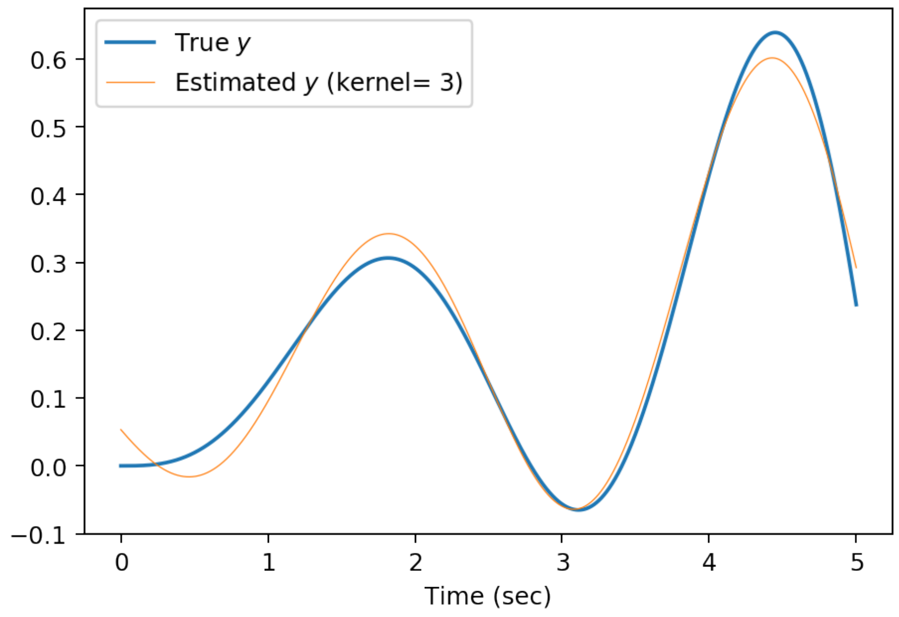

Figure 4.2 True y vs under-estimated y of order 3

It can be seen above that as we increase the model order, even for the same strategy of sampling at mid-point, calculating the kernels at said mid-point and then estimating the parameters to fit the non-linear curve. The estimation performs much better because the parameterization permits for higher dynamic range of the signal and there by allows for higher upwards and downwards slopes. Even though the order of our model and the true curve match in this case, the measured curve is Non-linear. While our model is linear, it can be seen that it still doesn't estimate the signal perfectly. When we increase the order of our model even further we observe the following trend:

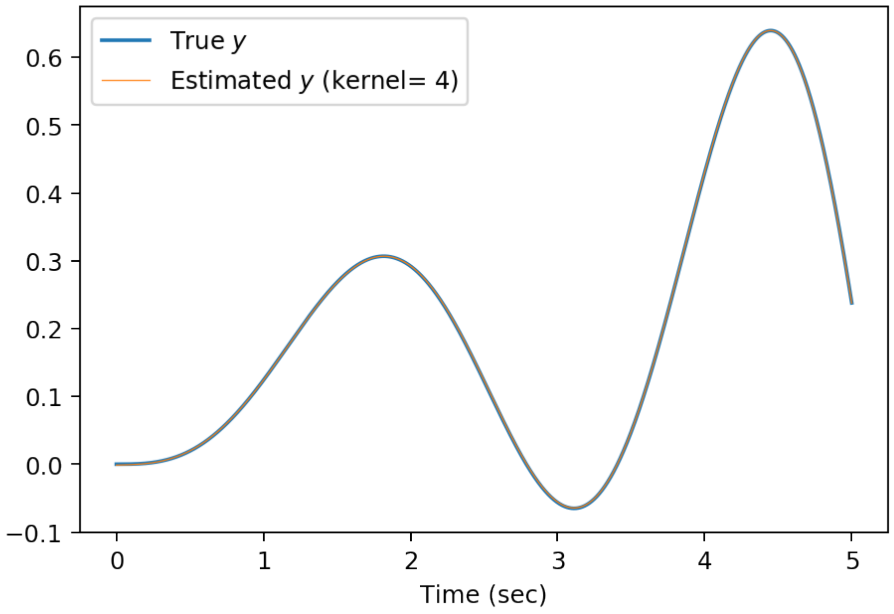

Figure 4.3 True y vs estimated y of order 4

From the above curve, it can be seen that our model does really well at estimating the measured original signal, so much so, that the two curves perfectly overlap even though the kernel was calculated only at the mid-point of the whole signal and the same estimated parameter values were used for estimating the whole curve over the entire horizon. In our trials with relatively simple non-linear systems like the one displayed here, one or two orders above the system order are capable of giving us good estimations. But what if we went one step further?

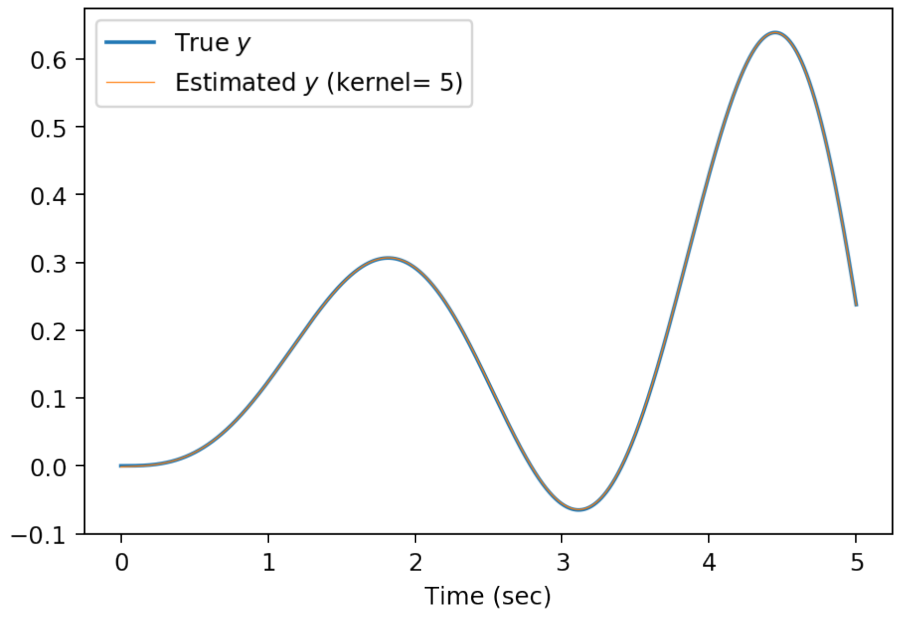

Figure 4.4 True y vs over-estimated y of order 5

The curve with an increased model order makes it clear to us in this case that it performs just as well as the 4th order curve and no better result than the one shown above can be formulated by tweaking any other parts of the estimation parameters. Furthermore, because it obtains the same result as the 4th order kernel case, we have an algorithm that has the cost of calculating an extra parameter for no improvement in performance. But, this superior performance in this case may not always be superior as explained in the previous section discussing bias-variance trade-off. Here, the model is fine-tuned to our curve which can be estimated with a 4th order kernel and above due to its dynamic nature. However, the problems start when the same algorithm is applied to a non-dynamic system and the problems with simply increasing the kernel order becomes apparent:

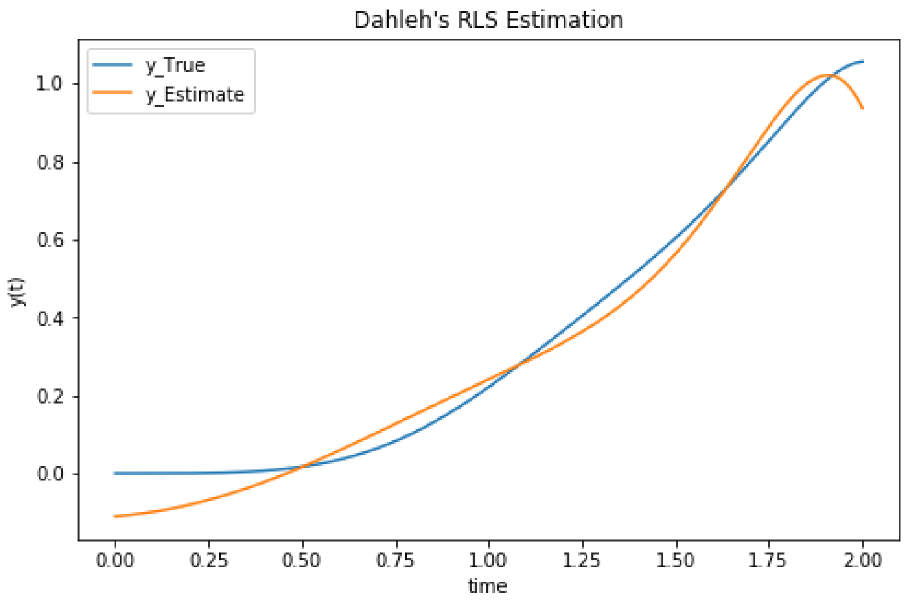

Figure 4.5 True y vs over-estimated y

The curve presented above shows a 2nd order signal being estimated by a $n = 4$ order kernel when sampled only at the center of the curve and as observed, even though the mid-point is estimated correctly, the estimated curve is far more dynamic than that of the original curve and which also results in in-efficient estimation which poses as a problem when the kernel order is much higher than required.

4.3 Motivation for Local Linearization and sliding window:

While, the above given signals are fairly smooth and can be easily estimated with just one realization of our linear model as long as the kernel order is hand-picked correctly on a case by case basis, the same observation does not extend to when the signal to be estimated is much more abrupt and rugged in nature for example:

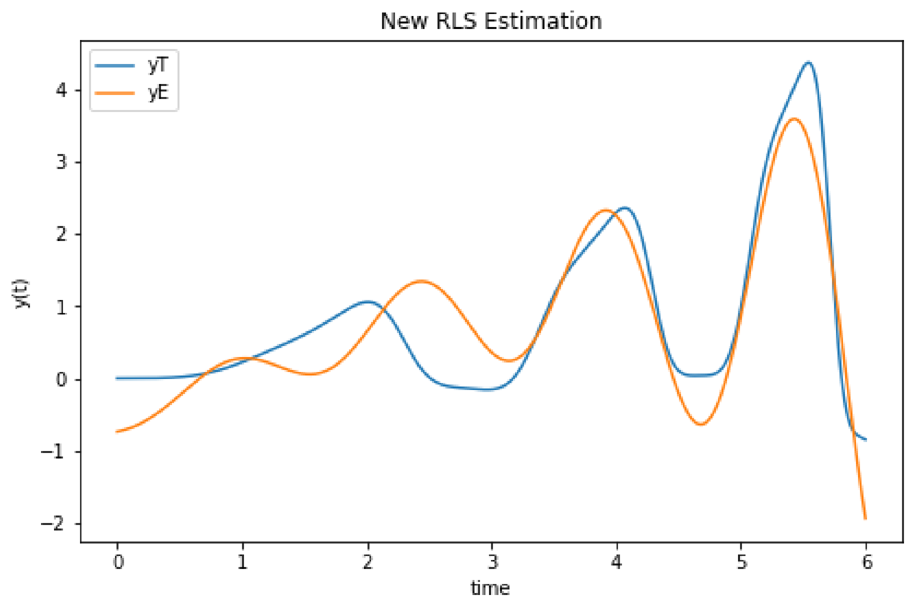

Figure 4.6 Abrupt non-linear signal y_{True} vs high order linear model $y_{Estimate}$

which is a 4th order non-linear system which has been estimated by a kernel of order $n = 4$ and as observed, performs poorly everywhere along the signal except for the middle, where it was sampled. This is what makes the non-linear domain of estimation all the more complex and exciting. Based on previous studies, there are several ways to move forward from this point [35], [36], [37], [38]. However, we apply a moving window approach [39] which will be explained in subsequent sections to estimate a non-linear curve.

4.4 Moving-horizon Output Estimation

The key idea of moving-horizon estimation is to divide the measured continuous Non-linear signal into p individual trajectories that will be estimated independently using a Linear estimator. Once all the sub-horizons are estimated, we will obtain p continuous curves which can be put together to obtain a large, more accurate estimation as opposed to estimating the whole curve at once. In principal, this is the same as estimating a Non-linear system output with a Linear time varying model.

The entire measured signal $y(t)$ is split into multiple batches of measurements, $[y_0, y_1, y_2, y_3, y_4...y_p]$. where

$$y(t_{window}) \subset y(t) \tag{4.11}$$

for all: $t_{window} \in t$ Such that,

$$y(t) = \sum_{i=0}^{P} (y_i(t_{window})) \tag{4.12}$$

where, t_{window} is a subsection of the time horizon over which $y(t)$ is defined, p is the total number of subsections of a window and N_p is the number of samples collected per batch and each individual trajectory section is treated as an independent estimation problem where we solve for (3.102), for a local linear model (2.2.1), then move on to the next window $(t + \Delta t)$ with the results of the previous window as initial conditions as explained visually below:

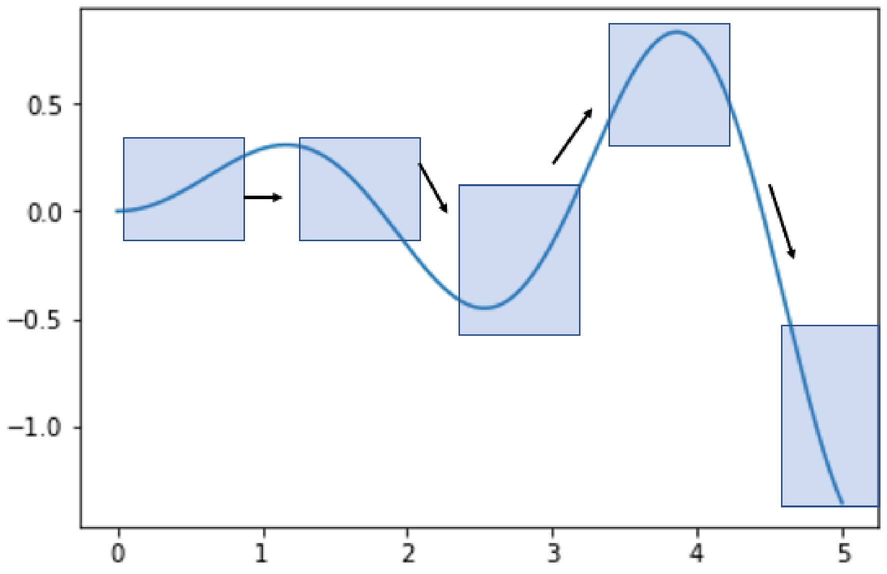

Figure 4.7 Window divisions for moving window estimate

The key variables to tune while implementing such a batch moving horizon estimator are:

- $Window - size$: The length of the time horizon that is to be included in each sub-window.
- $Window - shift(\Delta t)$: The amount of overlap between successive windows which also decides how quickly the window will move recursively across the entire signal.
- $Samples(N)$: The total number of samples that are collected in a sub-window.
- $kernel\ order(n)$: The order of the kernel and the local model to estimate the curve inside each window.

Consider the given algorithm and keep in mind the definitions given below:

- w_start: start of a sub-window.
- w_end: end of a sub-window.

- w_size: size of a sub-window.
- w_shift: The degree of shifting between consecutive windows.(Δ t)
- t_window: time horizon under a sub-window.
- Y_window: signal under a sub-window.
- T_end: End of the original signal

Algorithm 2 Calculate $Y_{estimate}$

```
initialize: sub-window
initialize:$a_k$
**while** $w\_end \leq T\_end$ **do**
    $w\_end = w\_start + w\_size$
    $Y\_window \leftarrow$ obtain $N$ samples under the window
    $t\_window \leftarrow$ obtain knots points under window
    $a_k, y \leftarrow$ RLS estimate using multiple regression using $a_{k-1}$
    $y_{estimate} \leftarrow$ append $y$
    $w\_start = w\_start + w\_shift$ (shift the window)
**end while**
**return** $y_{estimate}$
```

In the above given algorithm, it should be noted that the RLS algorithm is always initialized with parameters calculated in the previous window a_{k-1} since we expect the parameter values a_k to be in the vicinity of a_{k-1} or atleast evolve from it. This way, the whole algorithm can be improvised to work online as long as the data is contained in big enough batches over which the kernel integrates for noise rejection. Therefore, there is always a trade-off between the window-shift, window-size and accuracy of the estimated signal.
Consider the system given in (2.2.1), The diagram below depicts the original signal along with it's estimate with the moving window algorithm:

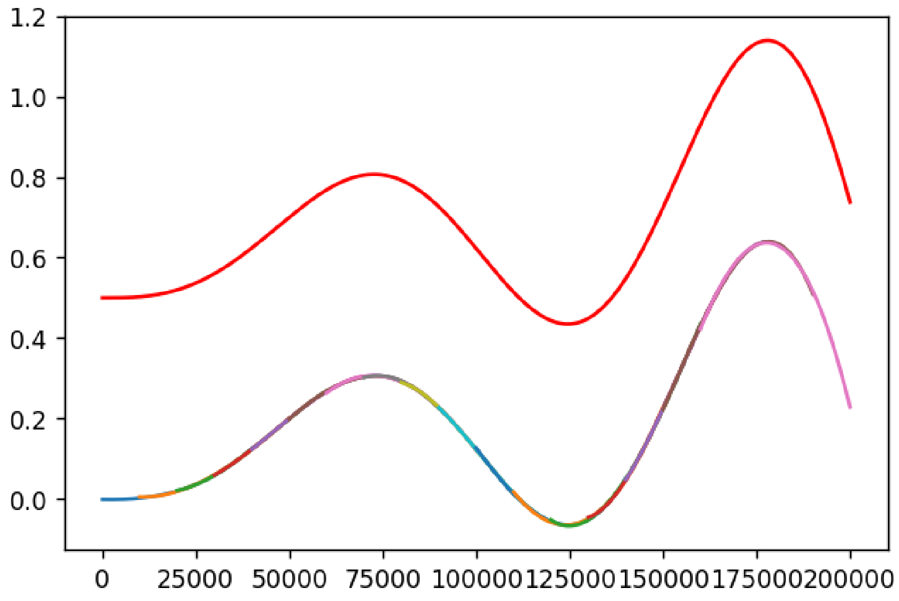

Figure 4.8 Estimation using moving window

In the above given diagram, the w_size was hard-coded to 0.5 seconds and each successive window had a half-window overlap. Hence, it is observed that different colours represent different sub-window estimates across a signal that has a total of $N = 200,000$ sample points and with a kernel order $n = 2$.

4.5 Over-parameterization under a window

As established previously in figure(4.3), when running RLS with the kernel order being higher than the system order we obtain accurate estimations, The same increment does not result in the better results in our moving window estimation algorithm.
The figure given below clearly shows the result of local linearizations of a signal when using higher order kernels ($n = 3$) which can help us achieve better noise rejection but result in offshoots at the beginning and end of the curve especially in cases where the only knot points where our kernels are calculated are at the mid-point of each window.

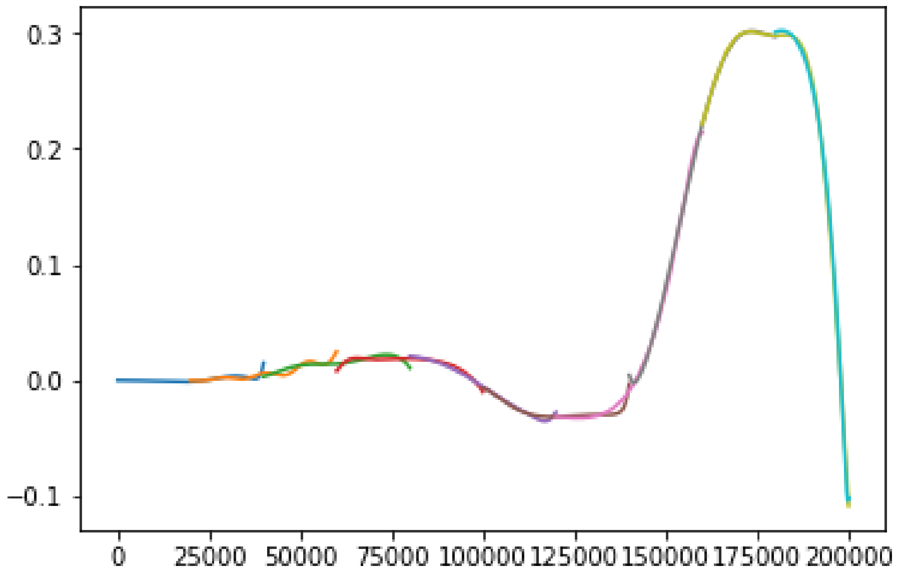

Figure 4.9 Overshoots in local linearization

Such offshoots are highly undesirable because they cause unrealistic gradient jumps when the entire signal is reconstructed by putting all the windows together with respect to (4.12). Further, with such offshoots, derivative reconstruction becomes extremely noisy(in practicez). While there are multiple ways of tackling this problem, three key candidates are:

- Changing the order of the kernel (n) while keeping the windows of fixed length resulting in an adaptive order moving window estimation. This method suffers from problems of initialization as some parameters in our linear model maybe added or subtracted depending on the signal in the window and subsequent windows suffering from improper initialization due to that change.

- Changing the window size (w_size) so as to have enough dynamic range within a window to ensure that the locally linear model is not over-parameterized for the current window which is the solution we explore in subsequent chapter and make further improvements on top.
- Changing the amount of overlap (w_shift) and more importantly, making sure that there is large overlap between subsequent windows which helps in descretization as then, we can only sample at the mid-point of the window and not worry about individual continuous estimated curves.

4.6 Results and discussion

In this section we will discuss the findings of our numerical simulation of the algorithms presented in the previous section and draw conclusions from their result.

4.6.1 RLS on Non-linear system

Since we subsequently moved on to appling our RLS algorithm to non-linear system, It is imperative that we conduct some analysis varying all the tunable parameters in our algorithm and look for positive correlation between estimation efficiency and any of the parameters, to gather insight into what parameters could be further developed to improve accuracy.
We consider the following system as our standard reference system across all experiments conducted in this section. Consider the following 3^{rd} order Non-linear time invariant system:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ a_0 x_1^3 + a_1 x_2^3 + a_2 x_3 \end{bmatrix} \tag{4.13}$$

where the measured output is,

$$y = x_1 \tag{4.14}$$

The constant parameters in this case are $a_0 = 1$, $a_1 = 5$,$a_2 = 5$. We estimate the signal with a 3^{rd} order kernel We initialize the system with the following initial conditions:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.15}$$

Moreover, The signal was sampled 200,000 times from which 3 knots were picked in all cases other than the ones that experiment with knots for calculating kernels for estimation. Where the signal spans from 0 to 6 seconds. Here is the true plot of the signal along with its estimation with a 3rd order kernel:

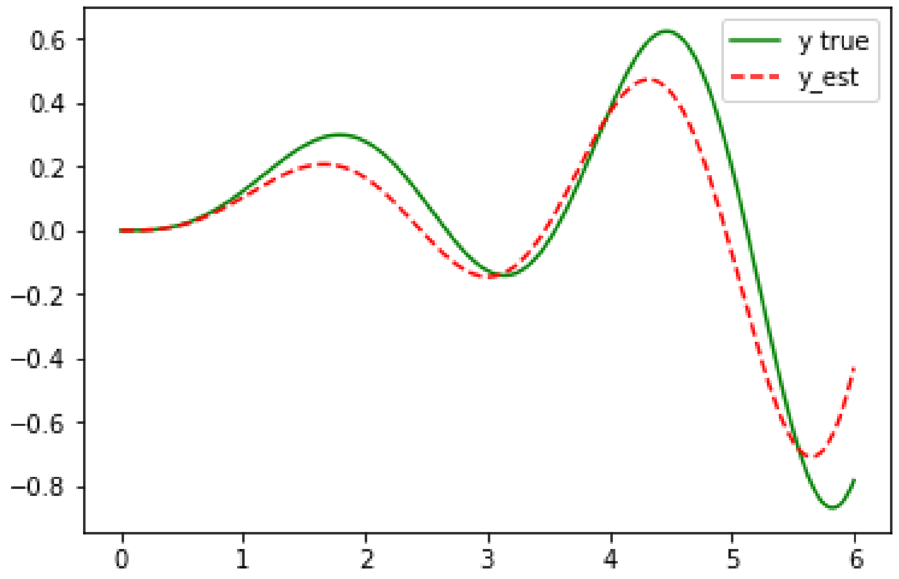

Figure 4.10 True non-linear curve and reconstructed with 3^{rd} order kernel with -5 dB noise

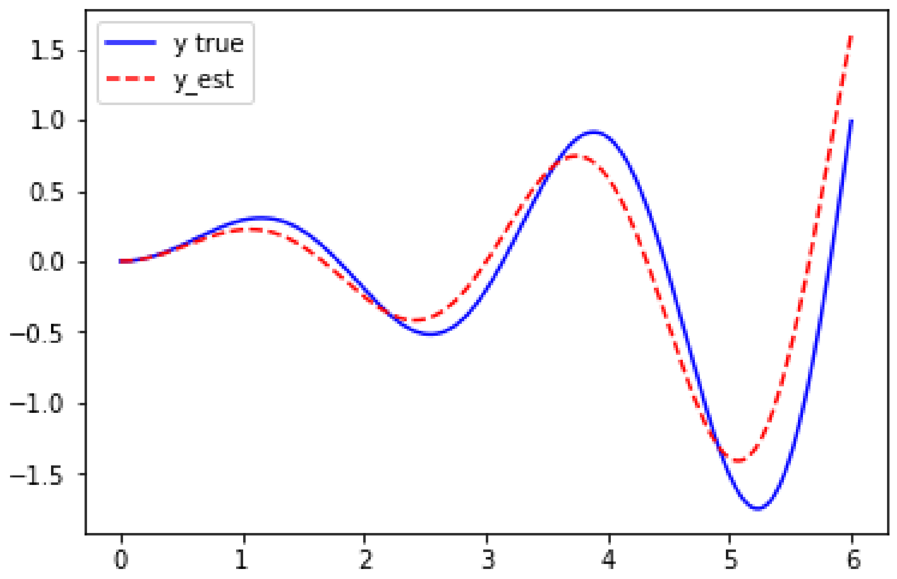

Figure 4.11 True non-linear first derivative and reconstructed with 3rd order kernel -5 dB noise

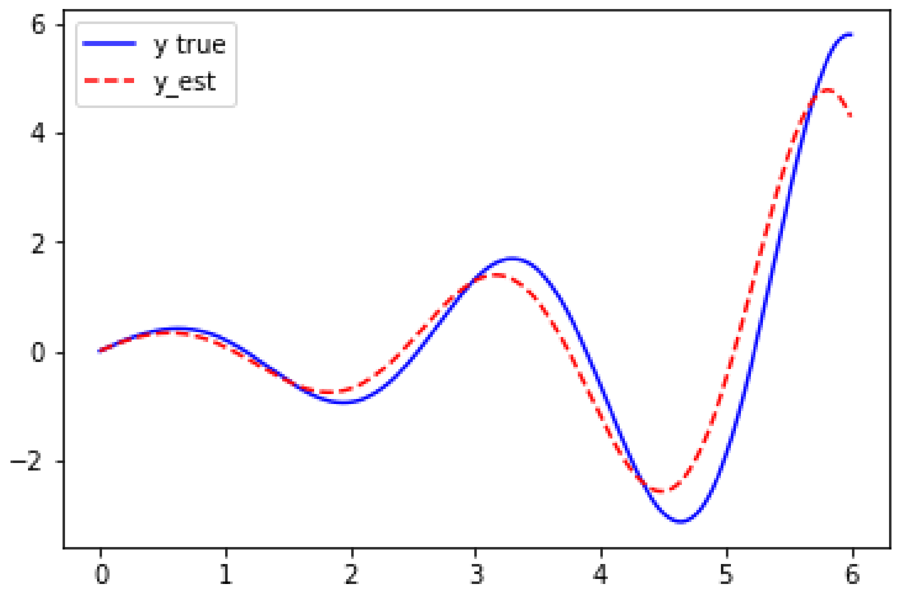

Figure 4.12 True Non-linear Second derivative and reconstructed with 3rd order kernel -5 dB noise

In the table below we examine the results for a 3^{rd} order kernel where "i-count" is the integrations of the reproducing property used for Multiple regression -

$Noise(dB)$	i-count	equation count	$knots$	$MAD(start)$	$MAD(mid)$	$MAD(end)$
-5.894	2-6	4	3	0.005385	0.005384	0.005382
-7.44656	2-6	4	3	0.006546	0.0065344	0.065425
-9.3873	2-6	4	3	0.031344	0.03133	0.03132
-11.8950	2-6	4	3	0.005637	0.005636	0.005636
-15.32156	2-6	4	3	0.065468	0.064554	0.05465
-19.85385	2-6	4	3	0.006503	0.006506	0.006509

Table 4.1 Effect of knot location on estimation for 3^{rd} order kernel

In the table below we examine the results for a 4^{th} order kernel for the same system as above

$Noise(dB)$	i-count	equation count	$knots$	$MAD(start)$	$MAD(mid)$	$MAD(end)$
-5.894	2-6	4	3	0.0168339	0.016838	0.016837
-7.44656	2-6	4	3	0.02513	0.02512	0.02511
-9.3873	2-6	4	3	0.03211	0.032156	0.032156
-11.8950	2-6	4	3	0.07986	0.07789	0.089663
-15.32156	2-6	4	3	0.05898	0.058961	0.0558938
-19.85385	2-6	4	3	0.113406	0.113367	0.11332

Table 4.2 Effect of knot location on estimation for 4^{th} order kernel

In the next table we compare the results for estimation of the same non-linear system using RLS estimation and ordinary least squares where the kernels as well as the MAD is calculated only at the mid-point, 3^{rd} order kernels were used here:

$Noise(dB)$	i-count	equation count	$knots$	$MAD(RLS)$	$MAD(OLS)$
-5.894	2-6	4	1	0.06021	0.05913
-7.44656	2-6	4	1	0.332010	0.30530
-9.3873	2-6	4	1	0.394788	0.08083
-11.8950	2-6	4	1	0.07986	0.07789
-15.32156	2-6	4	1	0.436408	0.015963
-19.85385	2-6	4	1	0.60769	0.058515

Table 4.3 RLS vs OLS at mid-point for a Non-linear system

In the table it can be observed that when applied only at one knot, the RLS reduces to an OLS estimator and therefore, their performance when evaluated only at one point does not change much. Going further we shall explore if the accuracy changes with other parameters. Next we explore the effect of integration count on performance of RLS and OLS at mid-point in a noiseless case:

$Noise(dB)$	i-count	equation count	$knots$	$MAD(RLS)$	$MAD(OLS)$
noiseless	2-6	4	1	0.35642	0.063543
noiseless	3-7	4	1	0.06343	0.07920
noiseless	4-8	4	1	0.05834	0.07288
noiseless	5-9	4	1	0.06886	0.11831
noiseless	6-10	4	1	0.07280	0.119556
noiseless	7-11	4	1	0.26462	0.068743

Table 4.4 RLS vs OLS at mid-point for a Non-linear system with changing integration count

The results don't seem to vary much in the above case, Next we vary the number of equations to see if there is any improvement:

$Noise(dB)$	i-count	equation count	$knots$	$MAD(RLS)$	$MAD(OLS)$
noiseless	2-6	4	1	0.35642	0.063543
noiseless	3-8	5	1	0.05885	0.06969
noiseless	2-8	6	1	0.06108	0.07397
noiseless	1-8	7	1	0.05727	0.05686
noiseless	1-10	9	1	0.08554	0.06586
noiseless	1-31	30	1	0.06613	0.06619

Table 4.5 RLS vs OLS at mid-point for a Non-linear with varying number of equations

Again, the change in performance when varying the number of equations above 4 doesn't trigger much change in the estimation because we already have 4 equations for 3 unknowns and we have more equations that what we need. One key parameter that we observe a positive correlation with estimation performance is number of samples in a trajectory. Below we consider two cases where in one case $N = 200,000$ and in another case $N = 2000$, for changing sizes of the overall trajectory to be estimated. Consider the following:

For the $N = 200,000$ case:

$Noise(dB)$	i-count	equation count	interval size	$MAD(RLS)$	$MAD(OLS)$
noiseless	2-6	4	0-8	0.08256	0.08652
noiseless	3-8	5	0-6	0.060215	0.05913
noiseless	2-8	6	0-4	0.027592	0.06361
noiseless	1-8	7	0-2	0.000185	0.00016

Table 4.6 RLS vs OLS at mid-point for a Non-linear with varying trajectory length $N = 200.000$

for sample size of $N = 2000$:

$Noise(dB)$	i-count	equation count	interval size	$MAD(RLS)$	$MAD(OLS)$
noiseless	2-6	4	0-8	0.092820	0.04254
noiseless	3-8	5	0-6	0.05758	0.05652
noiseless	2-8	6	0-4	0.04151	0.07257
noiseless	1-8	7	0-2	0.00352	0.00364

Table 4.7 RLS vs OLS at mid-point for a Non-linear with varying trajectory length $N = 2000$

The above two tables make it clear that when we have enough number of sample points concentrated in the interval over which the estimation is being conducted, the performance of our algorithm improves multiple fold in a noiseless case and this improvement is also seen in the noisy case where the kernel simple has a lot more samples to integrate over which even though takes much longer to run, does perform optimally. This result motivates us to apply this method over a sliding window so that we can exploit this increased accuracy over the entire signal instead of just the mid-point of the whole horizon as show in the next section.

4.6.2 Kernel based Moving-horizon estimation for a Non-linear system

In this section, we will discuss results of the moving horizon estimation technique when applied along with our kernels. Consider the following unstable Non-linear system with the following equations:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2^3 \\ x_3 \\ a_0 x_1 + a_1 x_2 + a_2 x_3 \end{bmatrix} \tag{4.16}$$

where the measured output is,

$$y = x_1 \tag{4.17}$$

The constant parameters in this case are $a_0 = 1$, $a_1 = 5$,$a_2 = 5$. we estimate the signal with a 3^{rd} order kernel We initialize the system with the following initial conditions:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.18}$$

Keep in mind that even though the parameters in our system in our system are constant, the parameters in our model will no longer be constant and be subject to change at every subsequent window because we are essentially fitting a collection of linear models by breaking up the entire signal over multiple windows. Therefore it can be said that we are estimating a Non-linear curve by a linear time varying system(LTV).

The system curves and it's derivatives are given below:

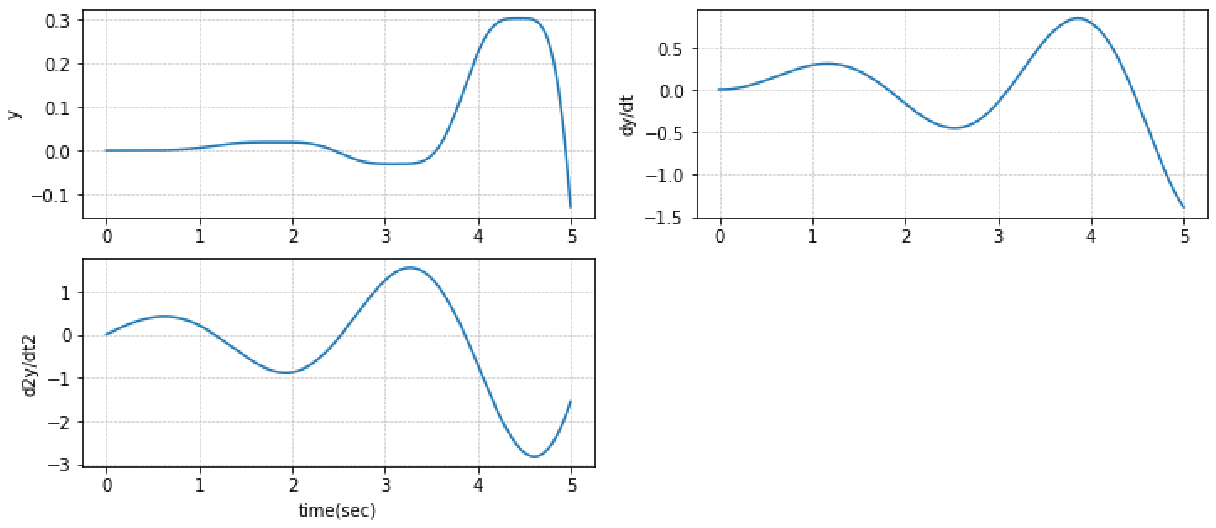

Figure 4.13 Reference for Non-linear system to be estimated by ltv system.

For our experiments with this systems the following parameters were kept constant:

- window size: 1 second
- window shift: 0.5 seconds
- integrations: 2-6
- kernel order: 3
- Number of samples: 200,000
- number of equations: 4
- knot position: mid and end point (2 knots per window)

Given below are the trajectories for increasing noise using our moving horizon estimation algorithm:

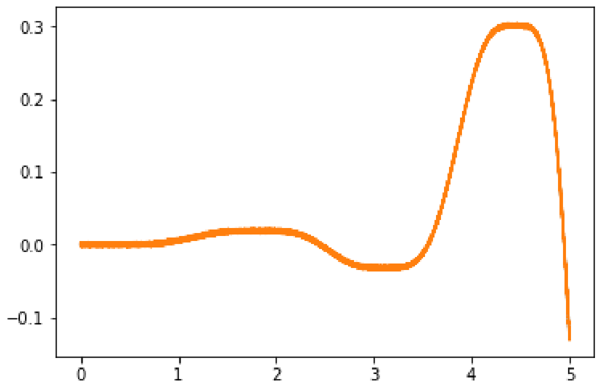

Figure 4.14 Reference trajectory with noise $\sigma = 0.001$

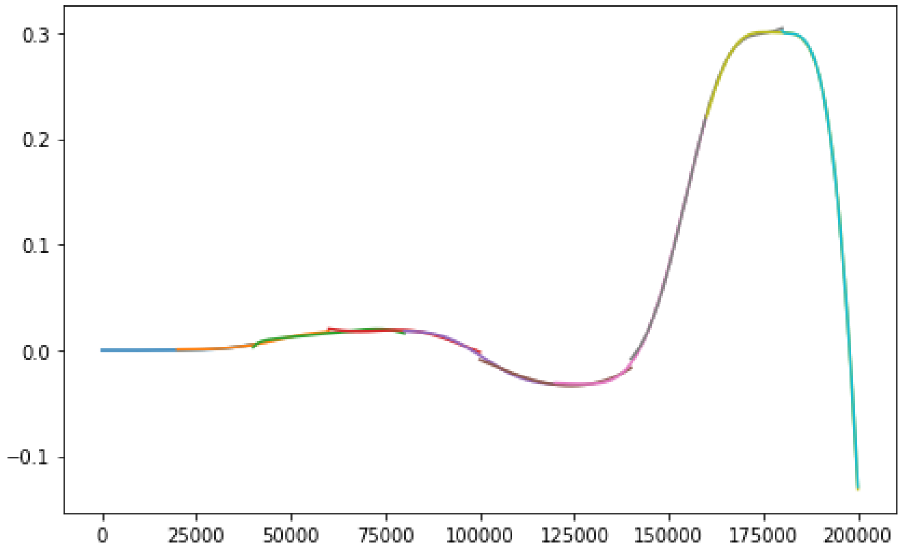

Figure 4.15 Reconstructed trajectory with noise $\sigma = 0.001$

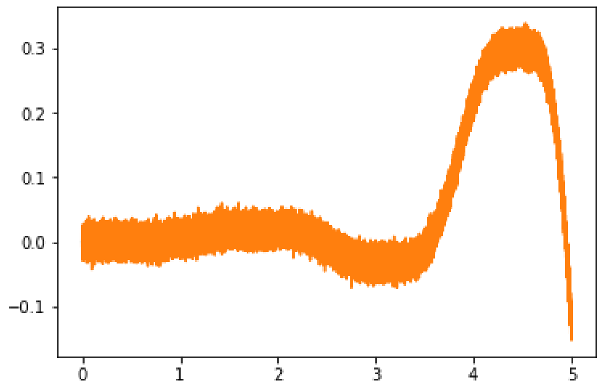

Figure 4.16 Reference trajectory with noise $\sigma = 0.01$

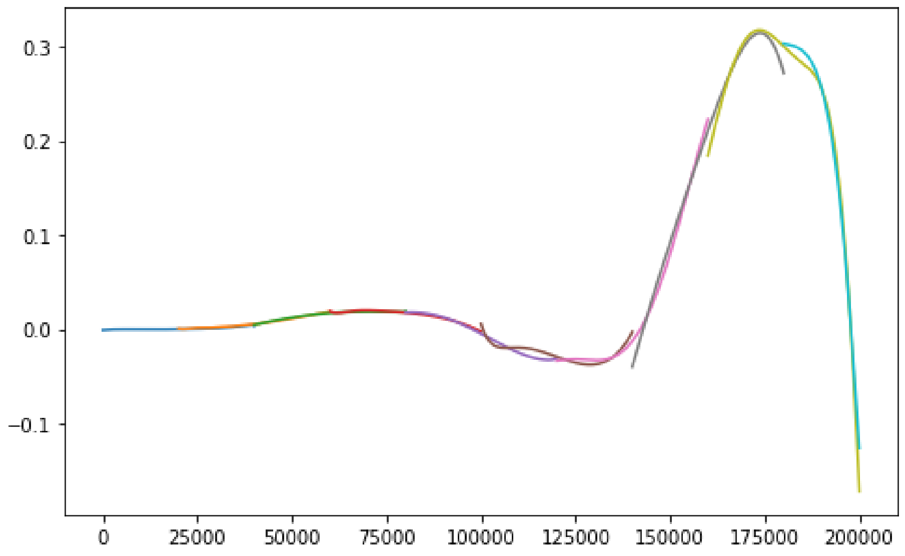

Figure 4.17 Reconstructed trajectory with noise $\sigma = 0.01$

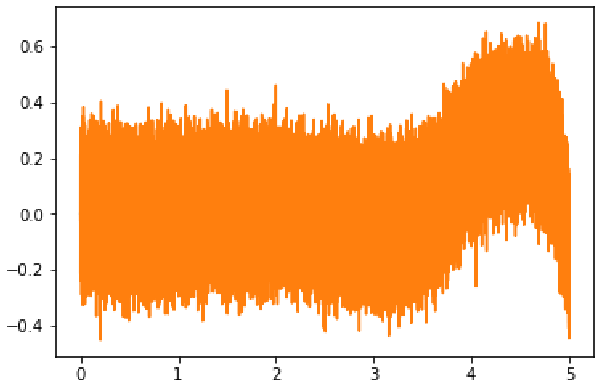

Figure 4.18 Reference trajectory with noise $\sigma = 0.1$

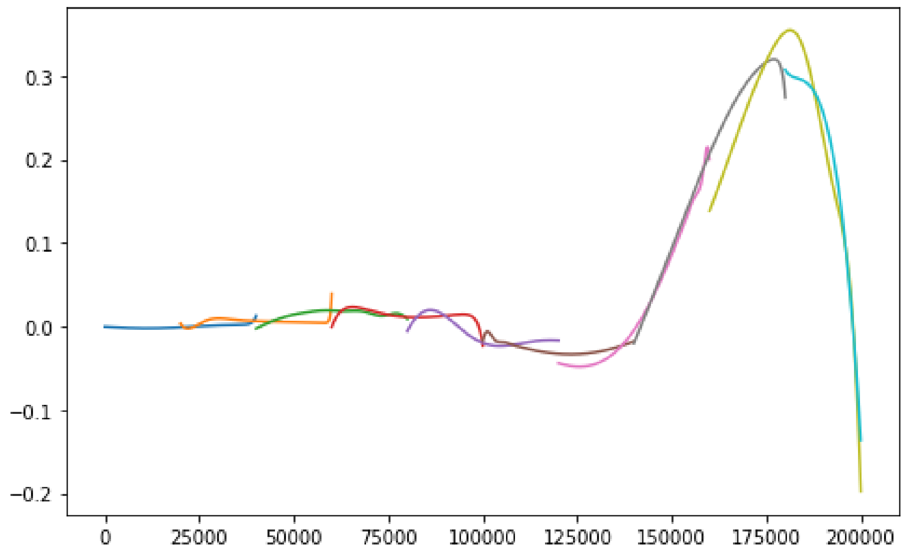

Figure 4.19 Reconstructed trajectory with noise $\sigma = 0.1$

Furthermore, we plot the parameters as functions of time for each case below:

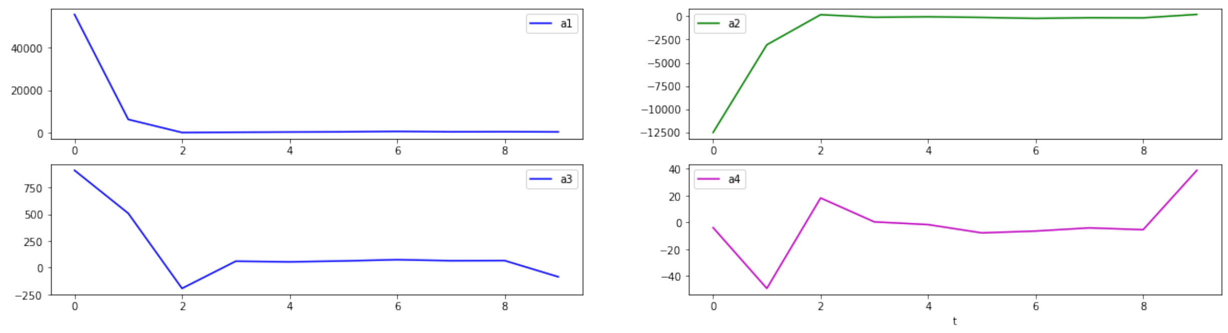

Figure 4.20 Parameters as function of time for $\sigma = 0.001$

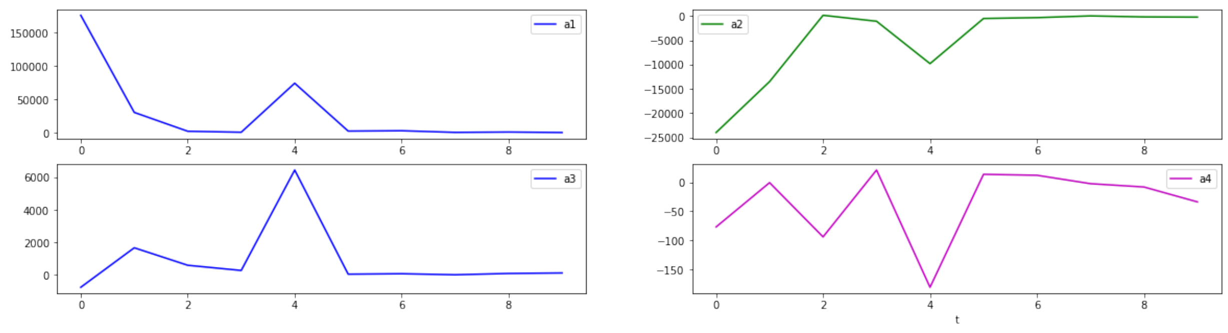

Figure 4.21 Parameters as function of time for $\sigma = 0.01$

It can be seen from the above cases that the parameters are far more dynamic in value as the noise increases, this is because the algorithm has far more uncertainty. and since we have a 4th order kernel, it has a tendency to over fit at each sub-window because of the over-parameterized nature of our estimation.

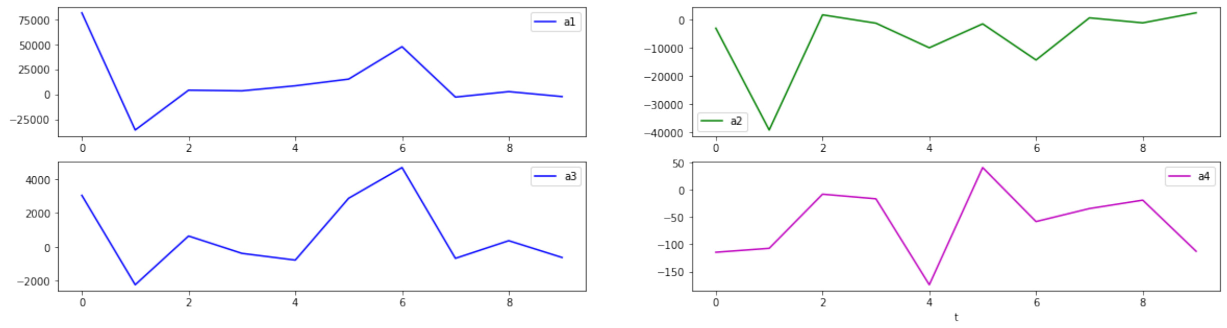

Figure 4.22 parameters as function of time for $\sigma = 0.1$

While the above results seem promising and one might be tempted to smoothen out the combination of smaller trajectories to form a bigger and yet smoother full trajectory, since there are such large offshoots especially in the case of us using a higher order kernel than the system, The derivative reconstruction severely fails to meet the standards even when working with low noise. Given below are the first and second derivative reconstruction for the first noise case of $\sigma = 0.001$.

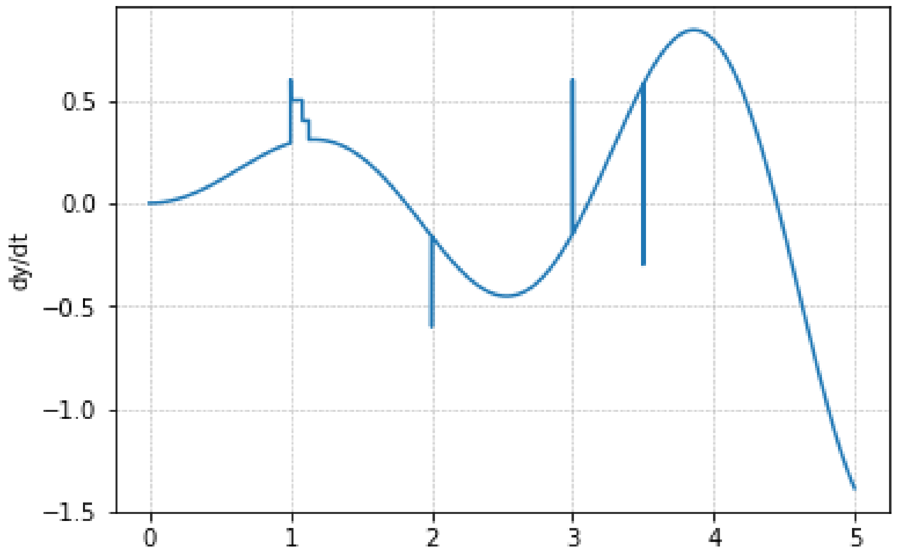

Figure 4.23 First derivative reconstruction using moving horizon with $\sigma = 0.001$

As observed in the plot above, The algorithm loses of the derivatives when there are

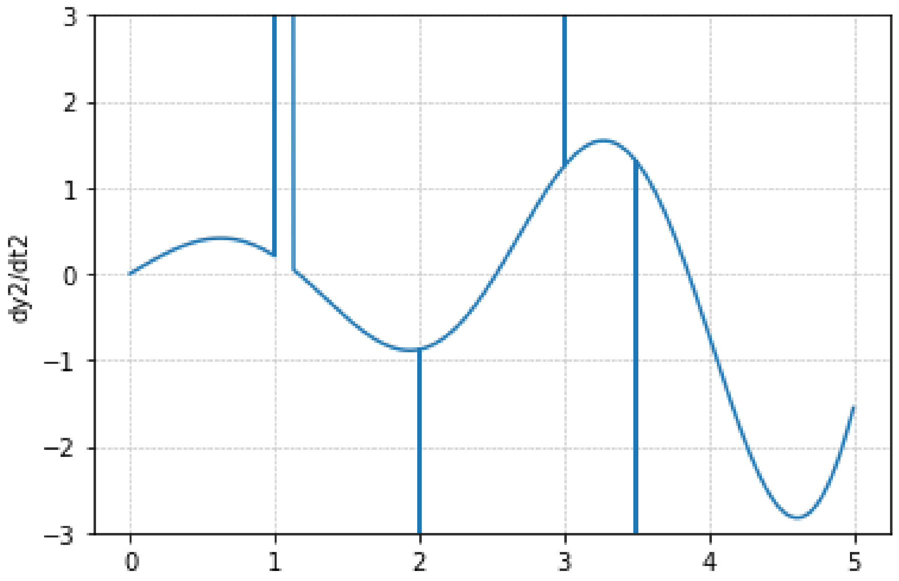

Figure 4.24 Second derivative reconstruction using moving horizon with $\sigma = 0.001$

sudden bumps in the estimated trajectory because essentially, slope at these points is uncertain or can't be measured accurately and as a result all subsequent derivatives thereof also see progressively worse spikes in their estimation. As we increase the noise higher, the uncertainty only gets worse and so does our derivative reconstruction.

Furthermore, we could improve the performance of our estimator by reducing the total number of subsections a given output trajectory is split into. Reducing the number of splits p would also result in reduced number of points in the curve where locally estimated curves need to be stitched together. Likewise, the next chapter deals with a technique in which the order of the locally linearizing system is kept higher and varying the window length in order to use lesser number of local curves.

Chapter 5

Adaptive Length Moving Horizon Estimator and Kernel Based Extended Kalman Filter

5.1 Adaptive length moving-window estimator

As discussed in the previous chapter, one important improvement to the moving-window estimator among a lot of contenders: [40], [41] [42], we would like to explore, is being able to dynamically increase or decrease the window length while estimating locally and also being able to exploit the noise rejection of higher order kernels. This is also done to ensure that the higher order kernel has enough signal to not be locally over-parameterized and end up with offshoots.

A Key metric to consider when implementing this increase or decrease mechanism is the *Dynamic range* of a signal as observed within each sub window as we traverse along the whole signal. A simple mathematical measurement for calculating the dynamic range is the peak-to-peak amplitude of a signal. Consider the following diagram:

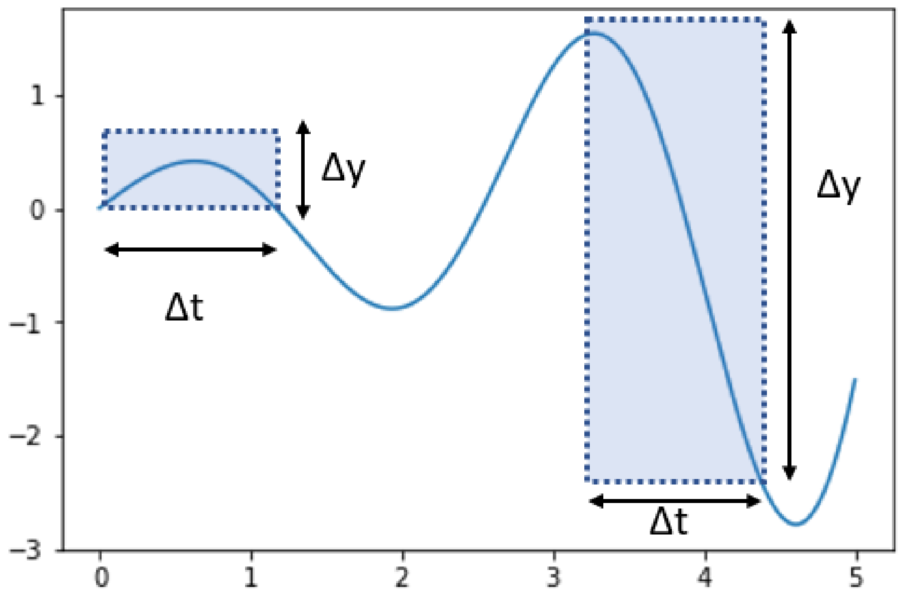

Figure 5.1 Dynamic range within a moving window

The two windows represented above have the same length in terms of their duration Δt but the corresponding range of values that Δy goes through within those two windows are considerably different, hence, using the same kernel order for both can result in over-parameterization in one or the other case as long as the window lengths remain fixed. In this thesis, we propose a method wherein, at the start of each window, when the algorithm obtains samples from within a window, we calculate the dynamic range such that:

$$dynamic\ range = |\mathbf{max}(y(t_{window})) - \mathbf{min}(y(t_{window}))| \tag{5.1}$$

using this dynamic range, we increase the window length as long as the dynamic range of the curve within the window is strictly increasing. The end of the window lies at the point where the dynamic range starts to decrease, thereby ensuring that a high dynamic range is covered within each window and each window starts and ends at either a crest or a trough of the signal to be estimated.
The algorithm for adaptive length moving window estimator is given below:

Algorithm 3 Calculate $Y_{estimate}$

```
initialize: $t_{window}$
initialize:$a_k$
while $w\_end \leq T\_end$ do
    $initialize$ : $d_k \leftarrow$ Dynamic range of current window
    while $d_k < d_{k+1}$ do
        if $d_k < d_{k+1}$) then
            $w\_size \leftarrow w\_size + \delta t_{window}$
        else
            $break$
        end if
    end while
    $w\_end = w\_start + w\_size$
    $Y\_window \leftarrow$ obtain $N$ samples under the window
    $t\_window \leftarrow$ obtain knots points under window
    $a_k, y \leftarrow$ RLS estimate using multiple regression using $a_{k-1}$
    $y_{estimate} \leftarrow$ append $y$
    $w\_start = w\_start + w\_shift$ (shift the window)
end while
return $y_{estimate}$
```

The above given algorithm works as long as the kernel order n of the local model is kept relatively high, approximately $n = 3$ so that enough change in gradient is presented within each sub window. consider the system given in 5.1, the following is the result of estimating the system given in (2.2.1):

As one can observe, the adaptive length moving window estimator with constant kernel

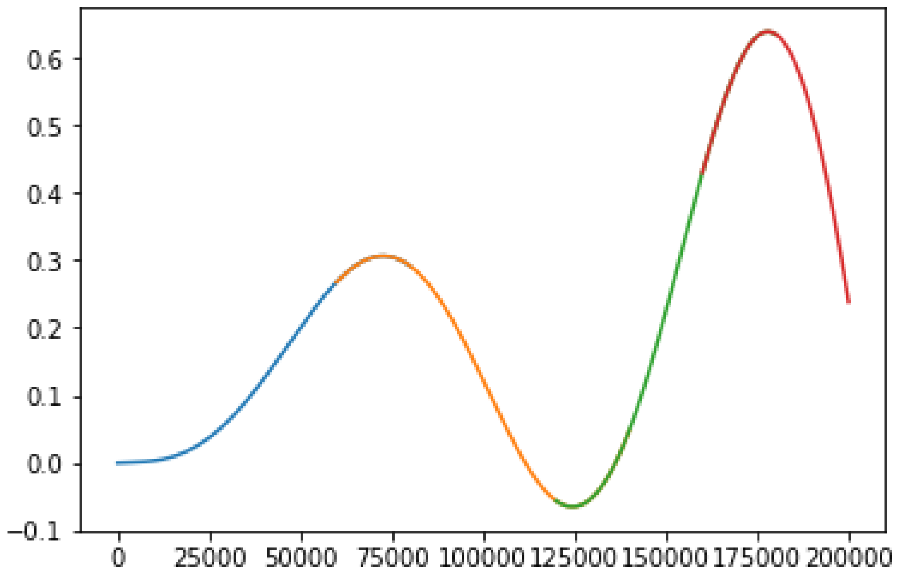

Figure 5.2 Reconstructed signal using adaptive window

order of $n = 3$, uses only 4 window shifts to estimate the system dynamics accurately. On the other hand, a fixed length estimator requires a lot more kernels to be calculated since it uses more windows and is also prone to over-parameterizing.

While the adaptive length moving window estimation has several benefits, a key question when estimating multiple smaller horizons is, how do you stitch these back together? One way to do this is by replacing the overlapping portion of a window estimate by the overlapping portion of the next estimate, this approach assumes that each subsequent estimate of a signal is better than the previous window which may not always be true. Another is to apply a simple numerical filter like the Savitzky-Golay filter [43] over the overlap of two windows. Even though this approach works on low noise cases, it suffers when we increase the noise and since we average over two windows and their overlap, if

one of the windows estimates the signal badly, the algorithm can end up propagating that error estimate throughout the estimation process.
These reasons make us realize the need for a way to correct estimations as we go through them in a window to window case which in turn motivates us to implement a prediction and correction model which can adapt to our uncertainty while simultaneously estimating the signal.

5.2 Results of simulation and discussion

Consider the following 3^{rd} order non-linear time invariant system:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ a_0 x_1^3 + a_1 x_2 + a_2 x_3 \end{bmatrix} \tag{5.2}$$

where the measured output is,

$$y = x_1 \tag{5.3}$$

The constant parameters in this case are $a_0 = 1$, $a_1 = 5$,$a_2 = 5$ and the rest of the hyper-parameters are the same as 4.6.1. In this section we will see the effect of having less overlap on each subsequent moving horizon affects the signal estimation, especially derivative reconstruction. In this case, even though the signal was estimated fairly well even with much lesser window shifts and a lot less kernel calculation, the limitations explained in the fixed window moving horizon estimation, still exist at the boundaries of each window where the slopes are uncertain and therefore the derivatives are calculated improperly.

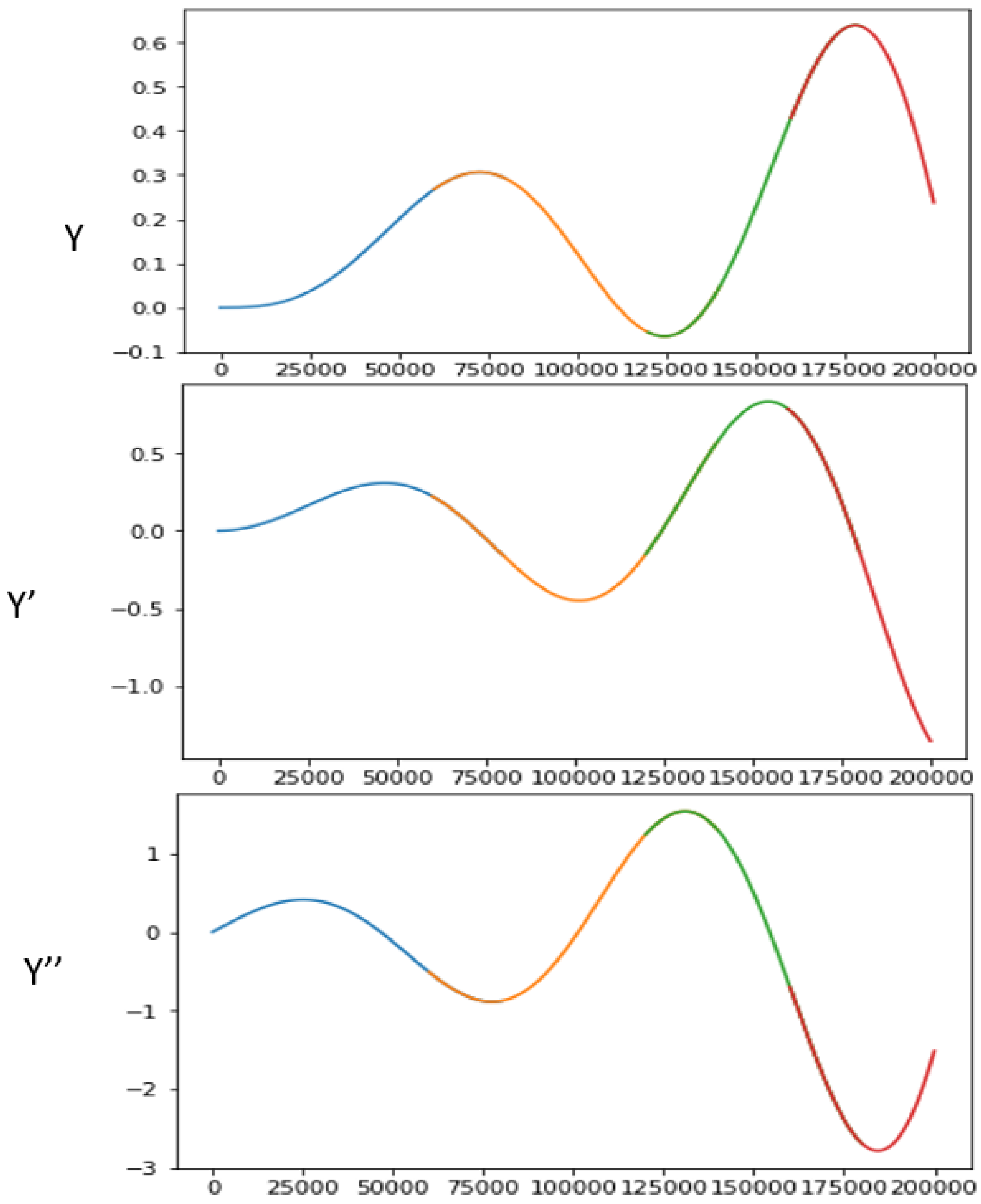

Figure 5.3 Signal and derivative reconstruction using adaptive length moving horizon with $\sigma = 0.0$

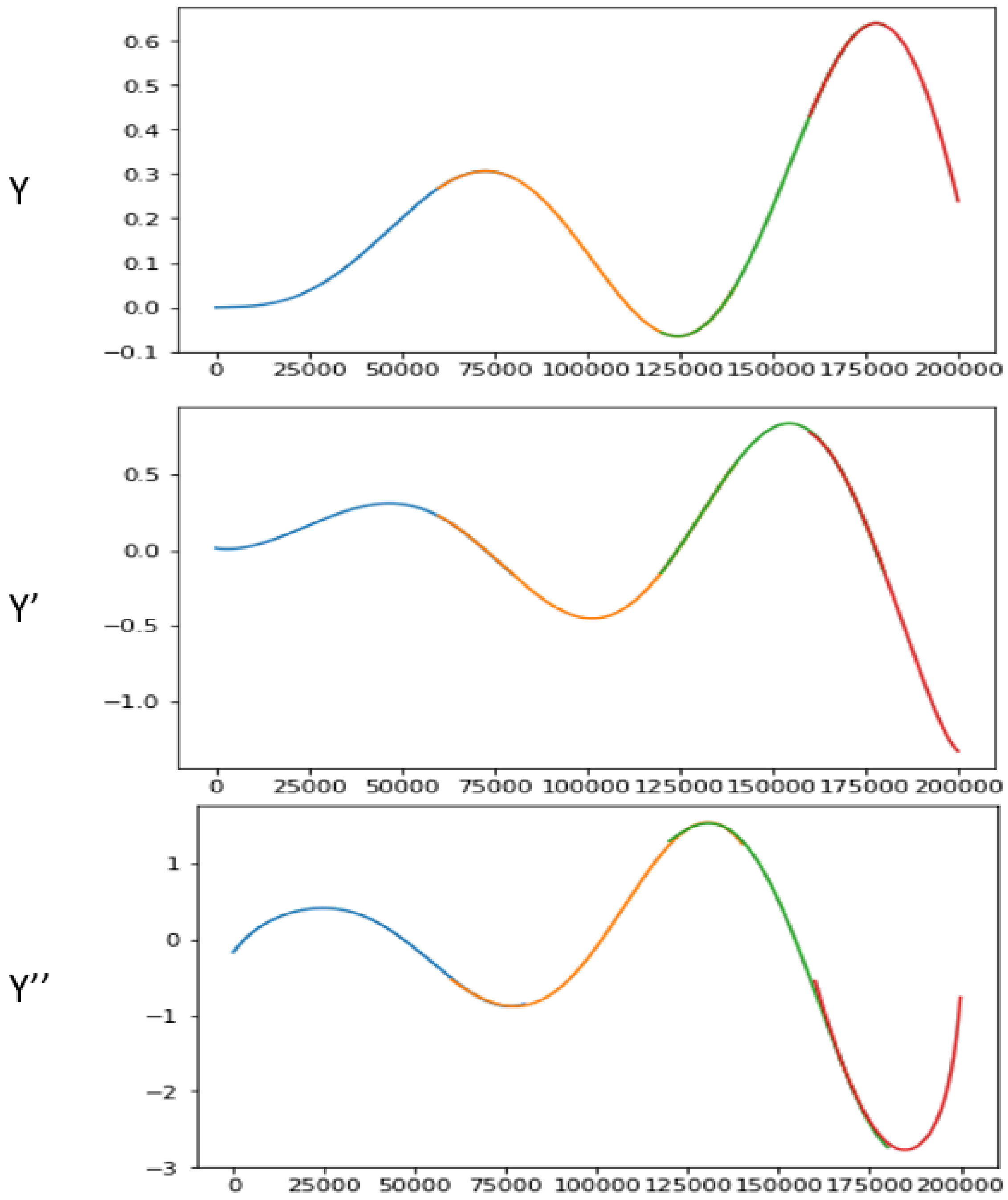

Figure 5.4 Signal and derivative reconstruction using adaptive length moving horizon with $\sigma = 0.1$

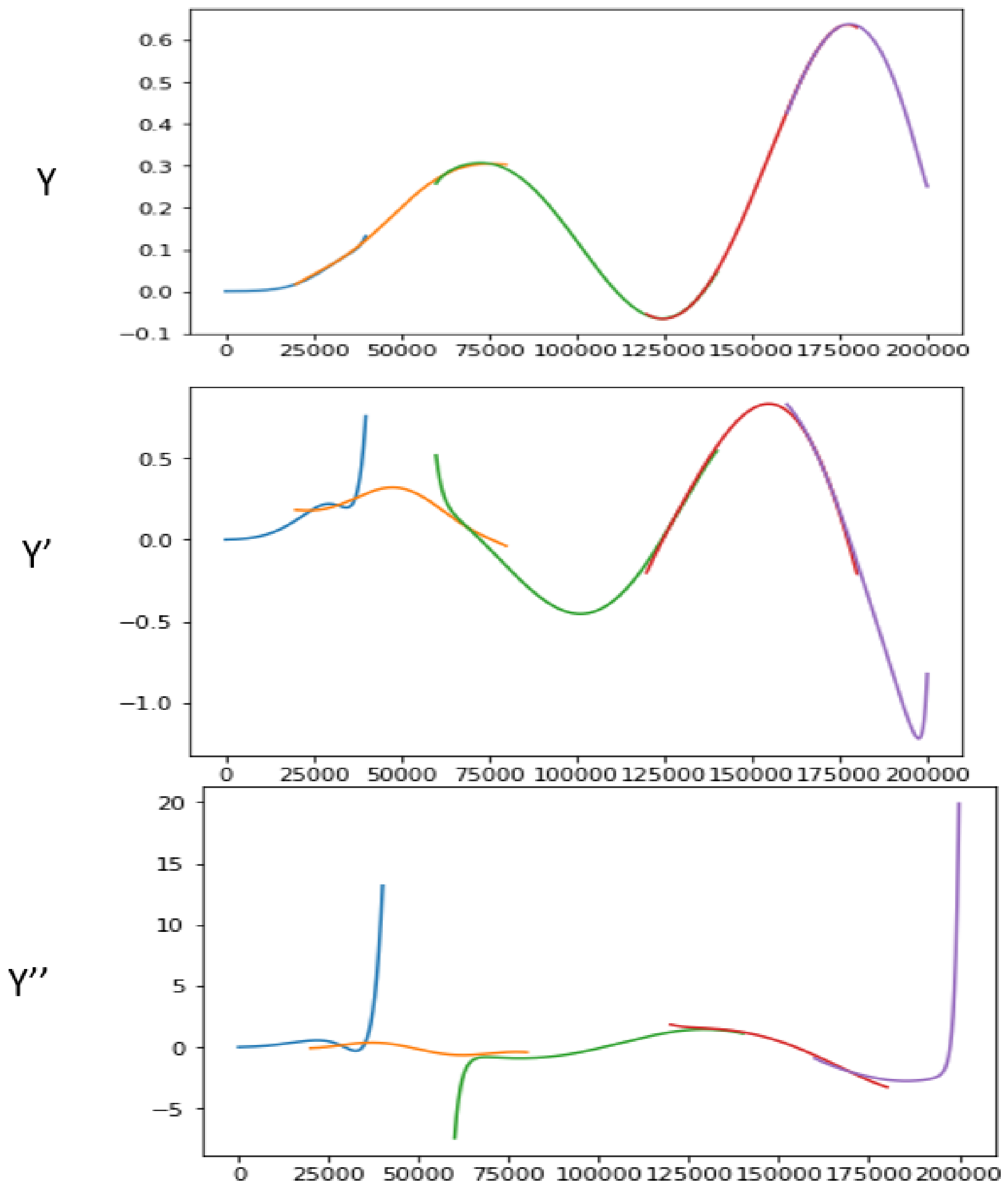

Figure 5.5 Signal and derivative reconstruction using adaptive length moving horizon with $\sigma = 1$

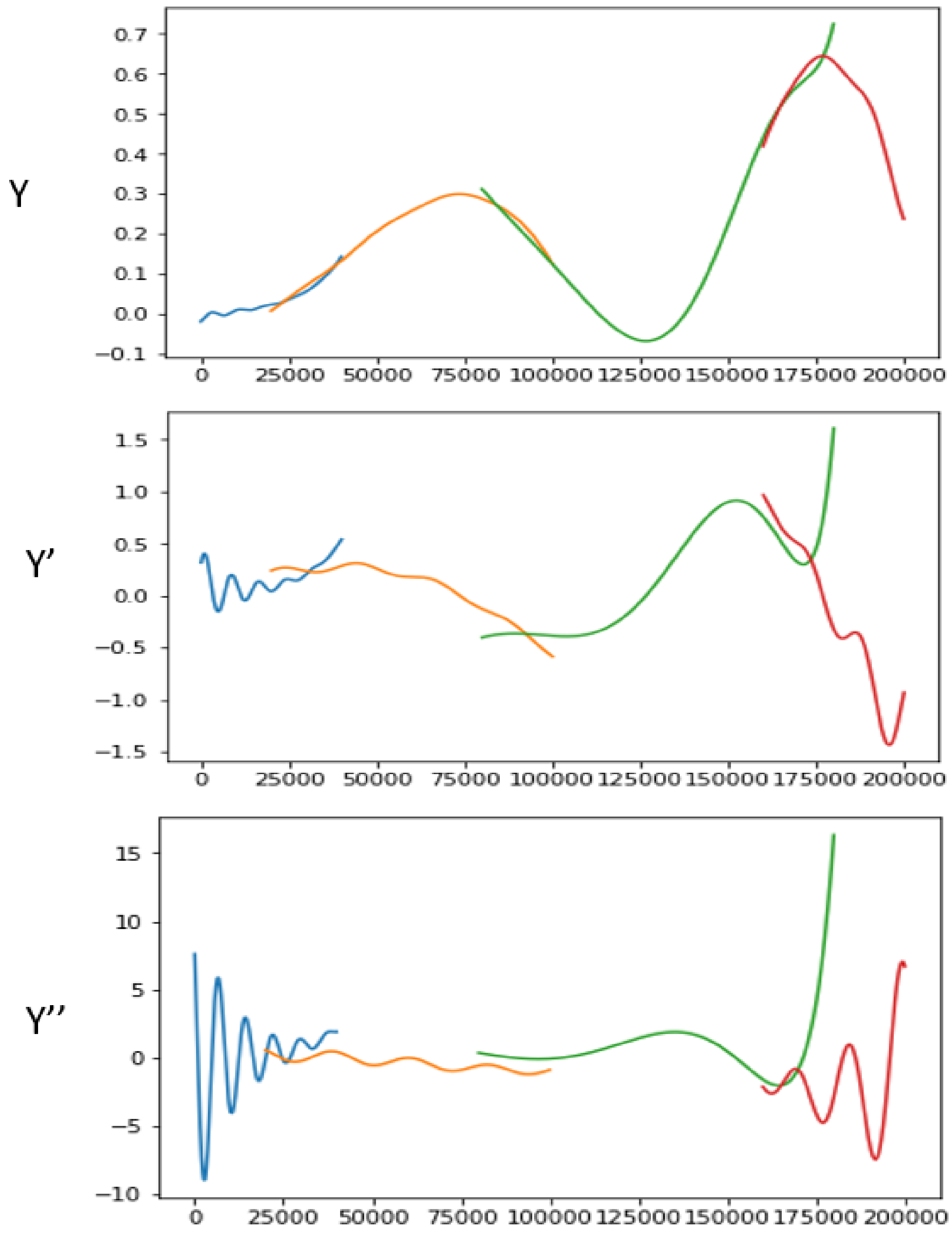

Figure 5.6 Signal and derivative reconstruction using adaptive length moving horizon with $\sigma = 5$

As observed from the graphs, as there is not much overlap between consecutive horizons compared to the size of each moving horizon, adding of higher noise, throws off the derivative reconstruction to the point where our performance is completely overwhelmed by noise. We experimentally determine that for all further trials of our algorithm from this point onward, must have a significantly larger window size, larger overlap between subsequent horizons, a relatively low kernel order ($n = 1$) as well as large number of sampling points per window. While all these conditions will make our method relatively slower compared to other algorithms, the trade-off is that we can handle noise much better even in the absence of model knowledge or initial conditions.

5.3 Motivation for discretization

From the results presented above and also those presented in the previous chapter, it is clear that when working with continuous curves, especially with high noise, "stitching" all the local linear curves together poses critical problems when trying to reconstruct the derivatives of the said curves. Case in point are the curves (5.5) and (5.6).
These conclusions motivate us to shift our focus towards local linearization in terms of discrete values instead of continuous smaller curves. In the following subsection, we only sample at the mid-point of each sub window and move to the next window. For this strategy to work, it is key that there is a large overlap between subsequent windows and that the mid point of each successive window is not too far from the previous. We will then be able to replace the local linearizing curves with their discrete mid-point values. A representative image is given below:

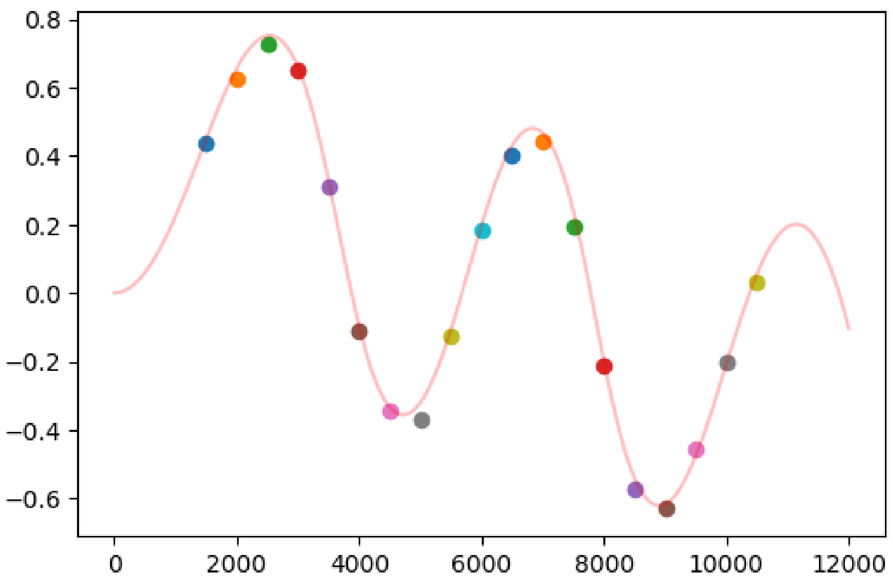

Figure 5.7 Discretization of local curves

where, each dot along the curve represents a mid-point of a locally linearizing curve. In practice, these dots will be much closer as will be seen in the results of the next algorithm.

5.4 Introduction to Kalman filter [8]

A Kalman filter [44] perfectly fits the bill for what we want to achieve given our short-comings in the previous section. It is an algorithm that takes in a series of measurements which in our case will be our batch of data,and produces estimates of unknown parameter-s/states that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each batch frame.

The algorithm works in two-step recursive workflow. In the prediction step, the Kalman filter produces estimates of the current state variables projected into the future along with their covariance uncertainties. Once a measurement is taken at the point at which we have forecasted, the Kalman filter re-conciliates the two by a relative weighted average, where either the measurement or the prediction may have added weightage based on the perceived measurement or prediction uncertainty that the filter observes.

The concept of state is fundamental in the Kalman filter. The state vector or simply state, denoted by $\mathbf{x}_i$, is defined as the minimal model that is sufficient to uniquely describe the unforced dynamical behavior of the system; the subscript i denotes discrete time. Typically, the state $\mathbf{x}_i$ is not observable. To estimate it, we use a set of observed data, denoted by the vector $\mathbf{y}_i$. The model can be expressed mathematically as:

$$x_{i+1} = F_i x_i + w_i \tag{5.4}$$

$$y_i = H_i x_i + v_i \tag{5.5}$$

where F_i is the transition matrix taking the state x_i from time i to time $i+1$, and H_i is the measurement matrix. The process noise w_i is assumed to be zero-mean, additive, white, and Gaussian noise, with the covariance matrix defined by

$$E[w_i w_j^T] = \begin{cases} S_i & \mathit{for}\ i = j \\ 0 & \mathit{for}\ i \neq j \end{cases} \tag{5.6}$$

Similarly, the measurement noise w_i is assumed to be zero-mean, additive, white, and

Gaussian noise, with the covariance matrix defined by

$$E[v_i \mathbf{v}_j^T] = \begin{cases} R_i & for\ i = j \\ 0 & for\ i \neq j \end{cases} \tag{5.7}$$

Hypothetically, a measurement on a linear dynamical system, described by (6.1) and (6.2) has been made at time i. The requirement is to use the information contained in the new measurement y_i to update the estimation of the unknown hidden state x_i. Let $\hat{x_{i-1}}$ denote a priori estimate of the state, which is already available at time i. In the algorithm of Kalman filter, the hidden state $\hat{x_i}$ is estimated as the linear combination of $\hat{x_{i-1}}$ and new measurement $\hat{y_i}$, in the form of:

$$\hat{x}_i = \hat{x}_{i-1} + K_i(y_i - H_i\hat{x}_{i-1}) \tag{5.8}$$

where matrix K_i is called Kalman gain. The full algorithm is shown in the following representation:

Where P^- represents the priori covariance and P represents the posterior covariance.

5.5 Kernel-based Extended Kalman Filter [9]

Since the focus of this thesis is to estimate Non-linear systems, we have a Non-linear dynamic system where you are not able to define either the process model or measurement model with multiplication of vectors and matrices as in (6.2) and (6.3). The extended Kalman filter provides us a tool for dealing with such nonlinear models in an efficient way. Since it is computationally cheaper than other nonlinear filtering methods such as point-mass filters [45] and particle filters [46], the extended Kalman filter has been used in various real-time applications like navigation systems.
The extended Kalman filter can be viewed as a nonlinear version of the Kalman filter that linearizes the model about a current estimate. The above formulation of the state and

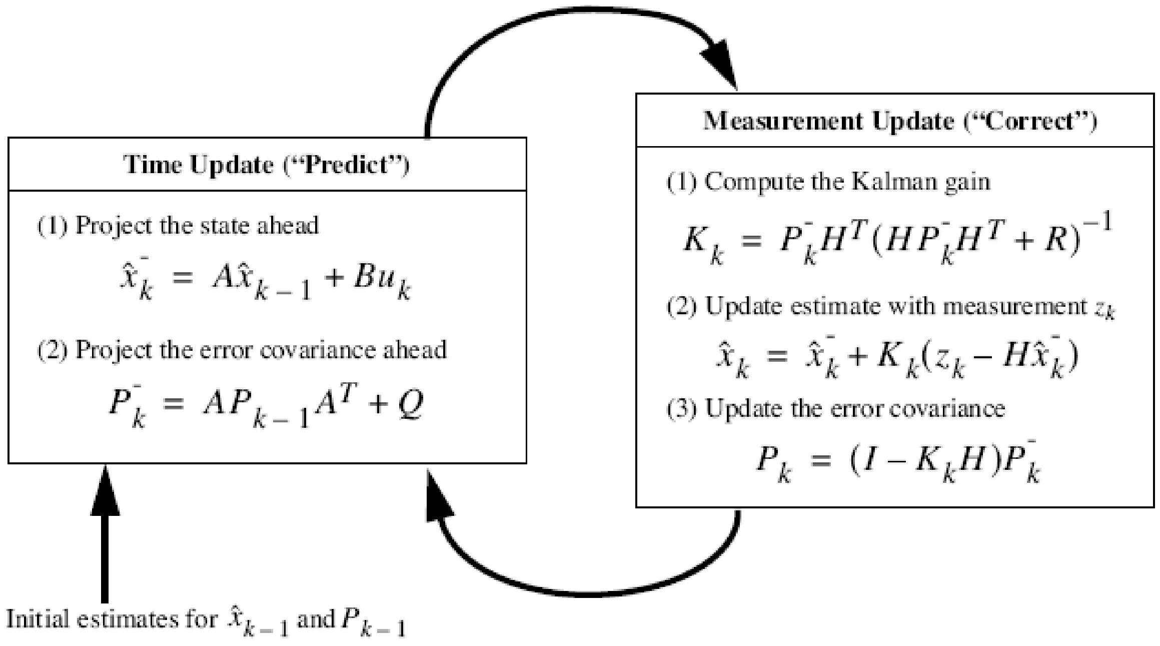
Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Figure 5.8 Recursive kalman filter flow chart

measurement equations can be updated as:

$$x_{i+1} = \mathbf{f}(x_i, u_i) + w_i \tag{5.9}$$

$$y_i = \mathbf{h}(x_i) + v_i \tag{5.10}$$

where $\mathbf{f}$ is the function of the previous state, x_{i-1} , and the control input, u_{i-1} , that provides the current state x_i . h is the measurement function that relates the current state, x_i , to the measurement y_i . w_i and v_i are Gaussian noise for the process model and the measurement model with covariance Q and R , respectively. The functions $\mathbf{f}$ and $\mathbf{h}$ are assumed to be C^1 functions, i.e, the function and it's first derivatives are continuous on the domain on which they are defined. In most practical applications of the extended kalman filter, the function $\mathbf{f}$ is approximated using a taylor series as:

$$\mathbf{f}(x_{i-1}) = \mathbf{f}(x^a_{i-1}) + \mathbf{J_f}(x^a_{i-1})(x_{i-1} - x^a_{i-1}) + \mathbf{H.O.T} \tag{5.11}$$

where $\mathbf{J_f}$ is the Jacobian of $\mathbf{f}(\cdot)$ and the higher order terms (**H.O.T.**) are considered negligible. Hence, the Extended Kalman Filter is also called the First-Order Filter. The Jacobian is defined as:

$$\mathbf{J_f} = \begin{bmatrix} \frac{\delta f_1}{\delta x_1} & \frac{\delta f_1}{\delta x_2} \cdots \frac{\delta f_1}{\delta x_n} \\ \vdots & \ddots \\ \frac{\delta f_n}{\delta x_1} & \frac{\delta f_n}{\delta x_2} \cdots \frac{\delta f_n}{\delta x_n} \end{bmatrix} \tag{5.12}$$

But since, we are not building a standard Extended Kalman filter rather an implementation that's based on our kernel, we modify the measurement function with our kernel based RLS implementation (3.5), and once we have the parameters (in our case $a_0, a_1, a_2...a_n$), we use the Taylor series expansion to then use those parameters and to predict the values in the next window as follows:

$$\mathbf{f}(x_i) = e^{A_{(i-1)}\Delta t} = I + A_{(i-1)}\Delta t + \mathbf{H.O.T} \tag{5.13}$$

Where $\mathbf{A}$ for a third order system would be:

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_{0_{i-1}} & -a_{1_{i-1}} & -a_{2_{i-1}} \end{bmatrix} \tag{5.14}$$

The approximation of $\mathbf{f}$ given above can sometimes result in inaccurate predictions and thus can be improved by superior Taylor series approximation methods such as the Pade approximation [47]. With the above mentioned changes, the rest of the kalman filter, in principle, works the same way as given in figure 5.8. Keeping in mind that in our implementation, we measure the signal only at the mid-point and predict only to the mid-point of the next window. But other sampling strategies can also easily be implemented.
The step-by-step algorithm for the same is given below (refer to the definitions given earlier):

Algorithm 4 Calculate $Y_{estimate}$ recursively

```
initialize: $t_{window}$
initialize:$a_{i-1}$
initialize:$\mathbf{P,Q,H,Rx}_{i-1}$
initialize:$w\_shift, w\_size, w\_end$
while $w\_end \leq T\_end$ do
    $w\_end = w\_start + w\_size$
    $calculate \mathbf{F}_i \leftarrow \mathbf{F}(a_{i-1})$
    Prediction Step:
    $\mathbf{x}_i = \mathbf{F}_i \mathbf{x}_{i-1}$
    $\mathbf{P}_i = \mathbf{F}_i \mathbf{P}_{i-1} \mathbf{F}_i^T + \mathbf{Q}$

    Kalman Gain update:
    $\mathbf{K}_i = \mathbf{P}_i \mathbf{H}_i^T [\mathbf{H}_i \mathbf{P}_i \mathbf{H}_i^T]^{-1}$

    innovation:
    $t\_window \leftarrow$ obtain knots points under window
    $a_i, \mathbf{z}_i \leftarrow$ RLS estimate using multiple regression with initial conditions:$a_{i-1}$
    $\mathbf{I}_i = \mathbf{z}_i - \mathbf{H}_i \mathbf{x}_i$

    Update Step:
    state update : $\mathbf{y}_i = \mathbf{K}_i \mathbf{x}_i$
    Covariance update:$\mathbf{P}_i = \mathbf{P}_i - (\mathbf{K}_i \mathbf{H}_i \mathbf{P}_i)$

    $y_{estimate} \leftarrow$ append $\mathbf{y}_i$
    $w\_start = w\_start + w\_shift$ (shift the window)
end while
return $y_{estimate}$
```

5.6 Results and discussion on Kernel based Extended Kalman Filter

For this final section we use another example of a 3^{rd} order non-linear system:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ a_0 x_1^{\frac{1}{2}} + a_1 x_2^2 + a_2 x_3 \end{bmatrix} \tag{5.15}$$

where the measured output is,

$$y = x_1 \tag{5.16}$$

The constant parameters in this case are $a_0 = 1$, $a_1 = 5$,$a_2 = 5$ Moreover, The following parameters are kept constant (wherever applicable) in the following results:

- Total number of samples **N**=200,000
- Trajectory length from [a, b] = [0,7] seconds
- kernel order **n**=3
- window length is 3 seconds
- window is shifted by 0.1 seconds
- subsequent window overlap is of 2.9 seconds
- kernel is evaluated only at mid-point of each window
- initial state = [1 , 0 , 2]

It is important to note that while our kernel based method works without any knowledge of the system dynamics and we don't need to know the state transition matrix, in the standard extended kalman filter implementation, we do need to know the system dynamics. In order to get the two implementations as close as possible in order to justifiably compare them, we will using an augmented system model as given in [33]. The state transition function in our comparison will look like the following for a 3^{rd} order system:

$$\mathbf{f} = \begin{bmatrix} x_2 \\ x_3 \\ a_0x_1 + a_1x_2 + a_2x_3 \\ a_0 \\ a_1 \\ a_2 \end{bmatrix} \tag{5.17}$$

where,

$$y = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{5.18}$$

Here we compare the trajectories obtained when implementing our kernel based extended filter (left) with the standard augmented system Kalman Filter(right) for changing noise:

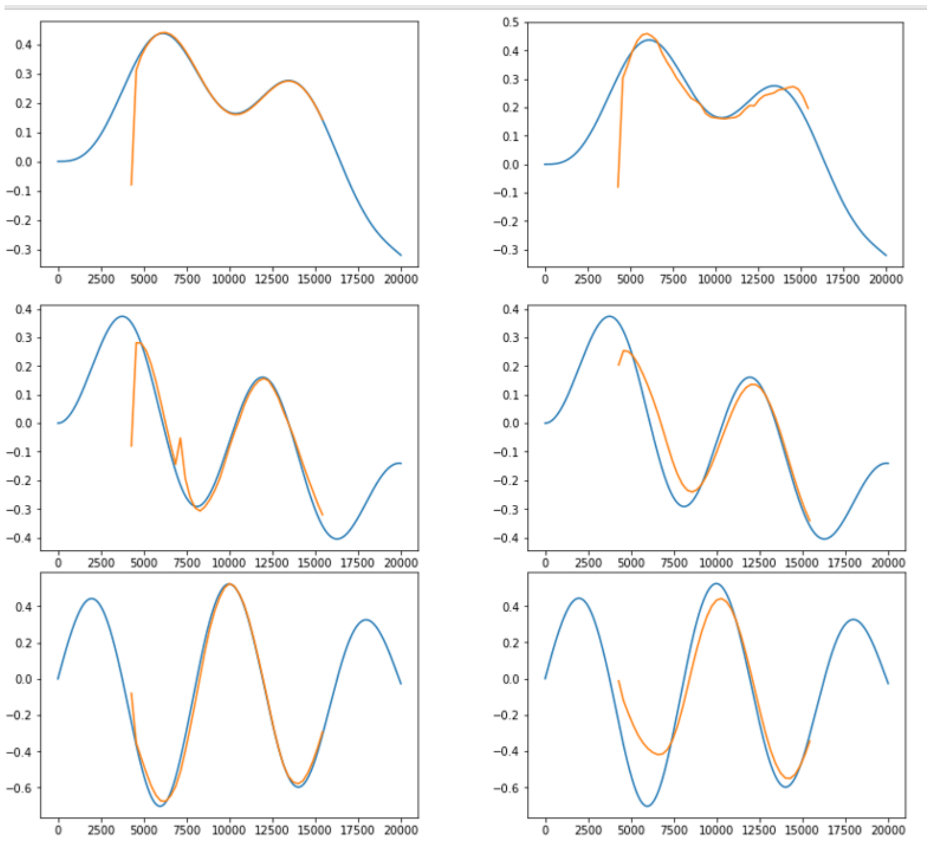

Figure 5.9 Kernel based EKF vs augmented EKF against ground truth with SNR = -5 dB

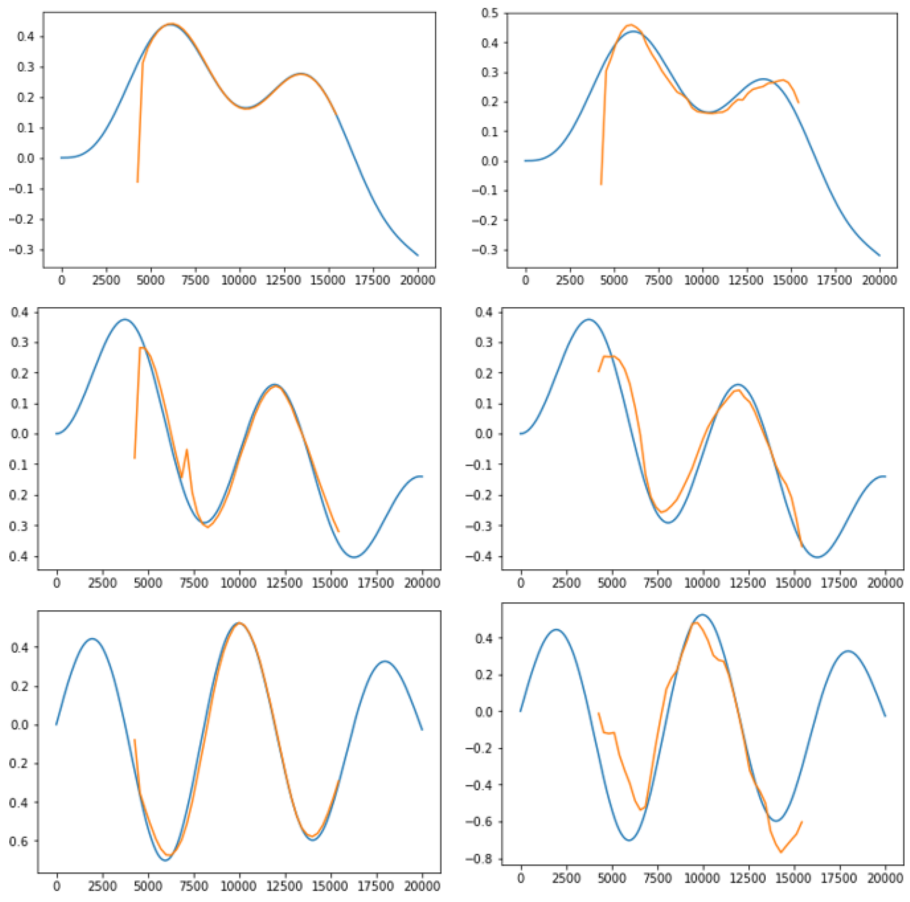

Figure 5.10 Kernel based EKF vs augmented EKF against ground truth with SNR = −8 dB

The table below summarizes the results obtained:

Algorithm	Noise	Output	MAD
Kernel -Extended kalman filter	σ=0.01	y	0.00356
		$y^{(1)}$	0.00121
		$y^{(2)}$	0.00334
Extended kalman filter	σ=0.01	y	0.00389
		$y^{(1)}$	0.003189
		$y^{(2)}$	0.002366
Kernel -Extended kalman filter	σ=0.05	y	0.05454
		$y^{(1)}$	0.04954
		$y^{(2)}$	0.06744
Extended kalman filter	σ=0.05	y	0.13456
		$y^{(1)}$	0.14351
		$y^{(2)}$	0.14995
Kernel -Extended kalman filter	σ=1	y	0.16645
		$y^{(1)}$	0.13256
		$y^{(2)}$	0.132553
Extended kalman filter	σ=1	y	0.36542
		$y^{(1)}$	0.46566
		$y^{(2)}$	0.26543

Table 5.1 Kernel based Extended kalman filter vs standard extended kalman filter

In the following graphs below, we purposefully distort the sampled trajectory with false entries which are out of the range of our noisy signal towards the middle of the trajectory to see if the algorithm gains back control of its tracking or completely loses its mark once it loses track of the original trajectory especially with noise:

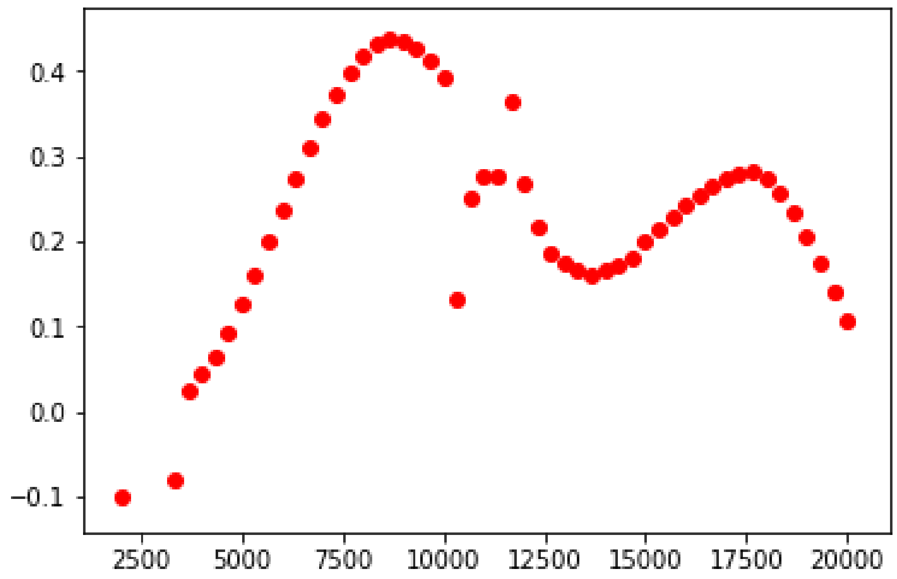

Figure 5.11 Estimation of distorted signal with noise $\sigma = 1$

As seen here, while the original signal recovers quickly, the subsequent derivatives take much longer to regain after such disturbance:

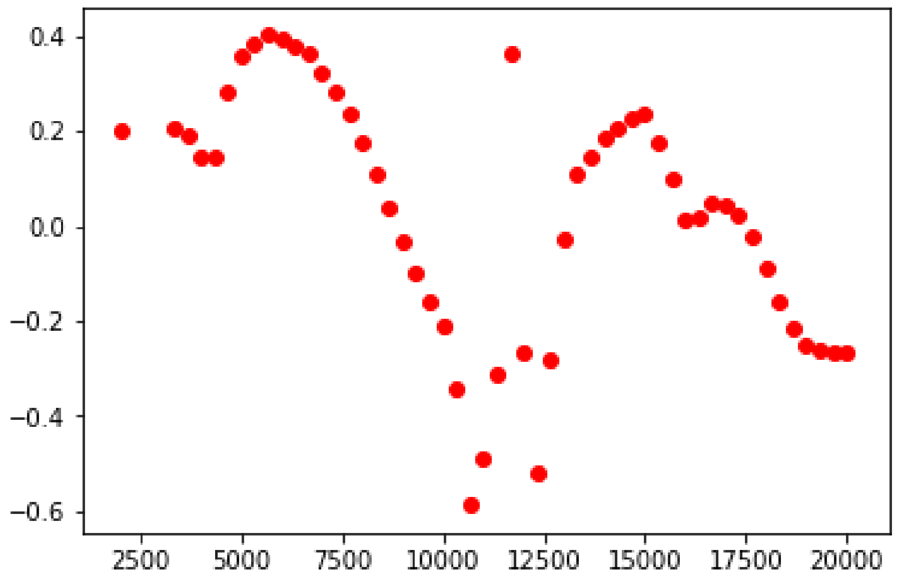

Figure 5.12 Estimation of first derivative of distorted signal with noise $\sigma = 1$

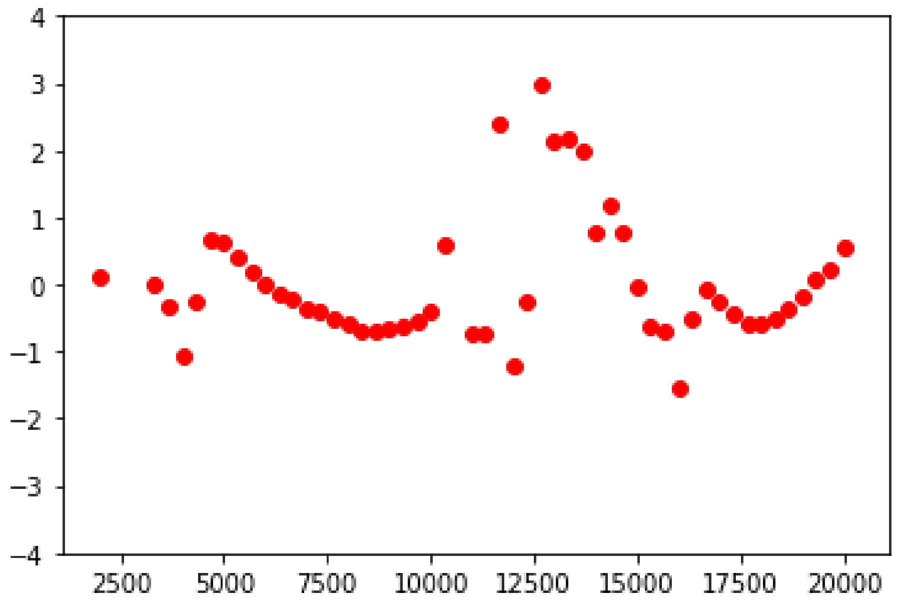

Figure 5.13 Estimation of second derivative of distorted signal with noise $\sigma = 1$

Chapter 6

Remarks and Future work

State and Parameter Estimation is of utmost importance to applications in almost all disciplines of science. The work done in this field is vast and diverse ranging from various classical methods [48], [49], [50] of parameter estimation, methods involving data assimilation [51], recursive stochastic filtering [11], [52] and many more. Recursive stochastic algorithms stand out due to their simplicity and efficient noise attenuating capability. This thesis makes significant contributions to work done in [53], [34], [26], and [54], by extending these methods to a linear time varying and non-linear domains. An important caveat to remember here is; even though the estimation results presented seem promising, they are subject to lots of parameters tuning, such as window length, window size, knots, sample size and so on. Nonetheless, our methods prove effective in tackling high noise and can be applied over a large range of systems, especially when tweaking the window size with the adaptive window length moving window estimator. The key parameters that were tuned for the kernel based kalman filter are number of samples per window, closeness of windows and window size, based on which we were able to obtain accurate estimations but at the cost of very high computational power and long processing times. Furthermore, covariance and linearizing state transition fuction were also tuned for better performance. Future works progressing this research should consider improving upon these elements of our models. The kalman filter can easily be extended to kernel based double-sided smoothing algorithms, applied to online estimation and even a particle filter. While, there exist infinite possibilities of what can be done, these suggestions would be important stepping stones to proceed further.

Bibliography

[1] DP Ghoshal, Kumar Gopalakrishnan, and Hannah Michalska. Algebraic parameter estimation using kernel representation of linear systems - submitted. In *IFAC World Congress, 2017. 20th World Congress*, 2017.

[2] Aritra Chatterjee. Two-stage kernel-based state and parameter estimation of lti systems. Master's thesis, McGill University, 2019.

[3] DP Ghoshal and Hannah Michalska. Forward-backward kernel-based state and parameter estimation for linear systems of arbitrary order. In *International Journal of Control*, (submitted).

[4] DP Ghoshal and Hannah Michalska. Finite interval estimation of lti systems using differential invariance,instrumental variables, and covariance weighting. In *American Control Conference, 2020. ACC 2020*, 2020 forthcoming.

[5] Shantanil Bagchi. Finite-time non-linear system identification and estimation using kernel-based adaptive sliding window. Master's thesis, McGill University, 2021.

[6] Mohammed Dahleh, Munther A Dahleh, and George Verghese. Lectures on dynamic systems and control. *A+ A*, 4(100):1–100, 2004.

[7] Anand Avati. Bias-Variance Analysis: Theory and Practice.

[8] Pingping Zhu, Badong Chen, and José C. Príncipe. Extended kalman filter using a kernel recursive least squares observer. In *The 2011 International Joint Conference on Neural Networks*, pages 1402–1408, 2011.

[9] Youngjoo Kim and Hyochoong Bang. Introduction to kalman filter and its applications. In Felix Govaers, editor, *Introduction and Implementations of the Kalman Filter*, chapter 2. IntechOpen, Rijeka, 2019.

[10] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. *Journal of basic engineering*, 83(1):95–108, 1961.

[11] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(69):35–45, 1960.

[12] D Luenberger. Observers for multivariable systems. *IEEE Transactions on Automatic Control*, 11(2):190–197, 1966.

[13] David G Luenberger. Observing the state of a linear system. *IEEE transactions on military electronics*, 8(2):74–80, 1964.

[14] Hebertt Sira-Ramirez, Carlos Garcia Rodriguez, John Cortes Romero, and Alberto Luviano Juarez. *Algebraic identification and estimation methods in feedback control systems*. John Wiley & Sons, 2014.

[15] Michel Fliess, Cédric Join, and Hebertt Sira-Ramírez. Non-linear estimation is easy. *International Journal of Modelling, Identification and Control*, 4(1):12–27, 2008.

[16] Hebertt Sira-Ramírez and Michel Fliess. An algebraic state estimation approach for the recovery of chaotically encrypted messages. *International Journal of Bifurcation and Chaos*, 16(02):295–309, 2006.

[17] Mamadou Mboup, Cédric Join, and Michel Fliess. A revised look at numerical differentiation with an application to nonlinear feedback control. In *The 15th Mediterrean Conference on Control and Automation-MED'2007*, 2007.

[18] Michel Fliess. Analyse non standard du bruit. *Comptes Rendus Mathematique*, 342(10):797–802, 2006.

[19] DP Ghoshal and Hannah Michalska. Finite interval kernel-based identification and state estimation for lti systems with noisy output data. In *American Control Conference, 2019. ACC 2019*, 2019 forthcoming.

[20] Kumar Vishwanath Gopalakrishnan. A new method of non-asymptotic estimation for linear systems. Master's thesis, McGill University, 2016.

[21] Hannah Michalska. Instructor Notes: Reproducing Kernels for Modelling of Systems, 2016.

[22] Lennart Ljung. *System identification: theory for the user*. Prentice-hall, 1987.

[23] DP Ghoshal, Kumar Gopalakrishnan, and Hannah Michalska. Using invariance to extract signal from noise. In *American Control Conference, 2017. ACC 2017*, 2017.

[24] DP Ghoshal, Kumar Gopalakrishnan, and Hannah Michalska. Parameter and state estimation in linear systems under arbitrary noise - submitted. In *IFAC World Congress, 2017. 20th World Congress*, 2017 forthcoming.

[25] Debarshi Patanjali Ghoshal, Kumar Gopalakrishnan, and Hannah Michalska. Kernel-based adaptive multiple model target tracking. In *IEEE Conference on Control Technology and Applications*, 2017.

[26] Surya Kumar Devarajan. Finite interval parameter and state estimationin lti systems using kernel-based multipleregression. Master's thesis, McGill University, 2020.

[27] Michel Fliess and Hebertt Sira-Ramírez. Closed-loop parametric identification for continuous-time linear systems via new algebraic techniques. In *Identification of Continuous-time Models from sampled Data*, pages 363–391. Springer, 2008.

[28] S Stanhope, JE Rubin, and David Swigon. Identifiability of linear and linear-in-parameters dynamical systems from a single trajectory. *SIAM Journal on Applied Dynamical Systems*, 13(4):1792–1815, 2014.

[29] Larry Schumaker. *Spline functions: basic theory*. Cambridge University Press, 2007.

[30] Norbert Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, 1949.

[31] Stefan Schaffler. *Generalized stochastic processes: modelling and applications of noise processes*. Springer Berlin Heidelberg, New York, NY, 2018.

[32] Wiener process, 2019.

[33] Debarshi Patanjali Ghoshal and Hannah Michalska. Forward-backward kernel-based state and parameter estimation for linear systems of arbitrary order, 2021.

[34] Anju John. Estimation for siso lti systems using differential invariance. Master's thesis, McGill University, 2019.

[35] K. P. B. Chandra and D. Gu. Jacobian-based nonlinear state estimation. In *None*, 2019.

[36] Mien Van, Hee-Jun Kang, and Y. Suh. A novel neural second-order sliding mode observer for robust fault diagnosis in robot manipulators. *International Journal of Precision Engineering and Manufacturing*, 14:397–406, 2013.

[37] S. Spurgeon. Sliding mode observers: a survey. *International Journal of Systems Science*, 39:751 – 764, 2008.

[38] N. Gordon, D. Salmond, and Adrian F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *None*, 1993.

[39] J.-j. E. Slotine, J. K. Hedrick, and E. A. Misawa. Nonlinear state estimation using sliding observers. In *1986 25th IEEE Conference on Decision and Control*, pages 332–339, 1986.

[40] Ziqi Zhang, Binqiang Xue, and Jianming Fan. Noise adaptive moving horizon estimation for state-of-charge estimation of li-ion battery. *IEEE Access*, 9:5250–5259, 2021.

[41] Marcel Paasche, Tor A. Johansen, and Lars Imsland. Regularized and adaptive nonlinear moving horizon estimation of bottomhole pressure during oil well drilling. *IFAC Proceedings Volumes*, 44(1):10511–10516, 2011. 18th IFAC World Congress.

[42] T. Kraus, H.J. Ferreau, E. Kayacan, H. Ramon, J. De Baerdemaeker, M. Diehl, and W. Saeys. Moving horizon estimation and nonlinear model predictive control for autonomous agricultural vehicles. *Computers and Electronics in Agriculture*, 98:25–33, 2013.

[43] Ronald W. Schafer. What is a savitzky-golay filter? [lecture notes]. *IEEE Signal Processing Magazine*, 28(4):111–117, 2011.

[44] Qiang Li, Ranyang Li, Kaifan Ji, and Wei Dai. Kalman filter and its application. In *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, pages 74–77, 2015.

[45] Jakub Královec and Miroslav Šimandl. Filtering, prediction and smoothing with point-mass approach. *IFAC Proceedings Volumes*, 37(6):375–380, 2004. 16th IFAC Symposium on Automatic Control in Aerospace 2004, Saint-Petersburg, Russia, 14-18 June 2004.

[46] Simon Godsill. Particle filtering: the first 25 years and beyond. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7760–7764, 2019.

[47] Hassan Ismail, I. Youssef, and Tamer Rageh. New approaches for taylor and padé approximations. *International Journal of Advances in Applied Mathematics and Mechanics*, 2:78–86, 03 2015.

[48] Lennart Ljung. System identification: Theory for the user, ptr prentice hall information and system sciences series. *ed: Prentice Hall, New Jersey*, 1999.

[49] Karl Johan Åström and Peter Eykhoff. System identification—a survey. *Automatica*, 7(2):123–162, 1971.

[50] GC Goodwin and KS Sin. Adaptive filtering prediction and control. 1984. *Englewood Clifs: Prentice Ha lI*, 1984.

[51] Eugenia Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.

[52] Thomas Kailath, Ali H Sayed, and Babak Hassibi. *Linear estimation*, volume 1. Prentice Hall Upper Saddle River, NJ, 2000.

[53] Debarshi Patanjali Ghoshal, Kumar Gopalakrishnan, and Hannah Michalska. Using invariance to extract signal from noise. In *American Control Conference (ACC), 2017*, pages 2588–2593. IEEE, 2017.

[54] Farhad Rahbarnia. Semi-deterministic finite interval estimation of linear system dynamics and output trajectory. Master's thesis, McGill University, 2020.