Continuous Time Identification of Systems with known and unknown order

Varanasi Santhosh Kumar

A Thesis Submitted to

Indian Institute of Technology Hyderabad

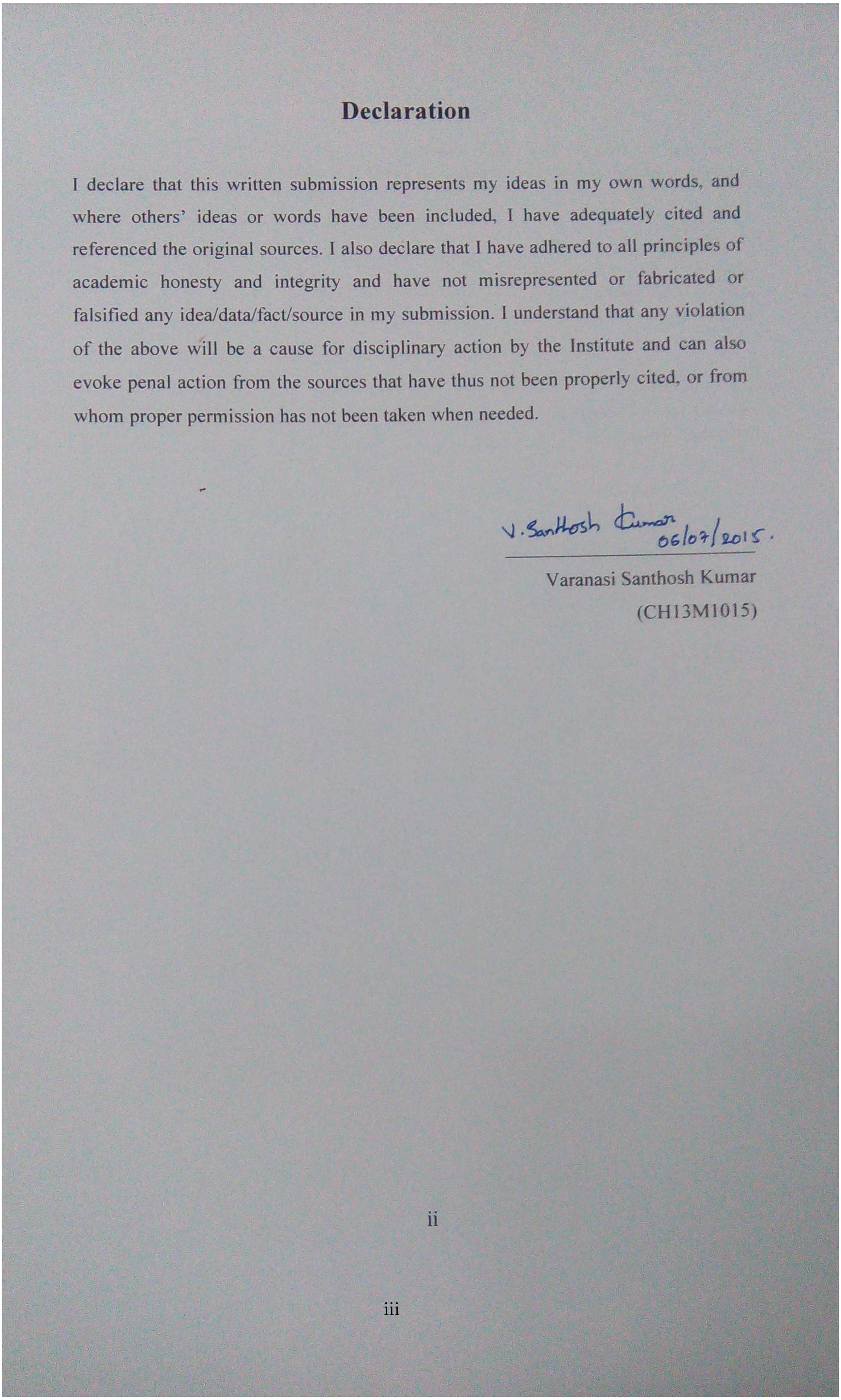
In Partial Fulfillment of the Requirements for

The Degree of Master of Technology

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Department of Chemical Engineering

June 2015



Declaration

I declare that this written submission represents my ideas in my own words, and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.

V. Santhosh Kumar
06/07/2015.

Varanasi Santhosh Kumar
(CH13M1015)

Approval Sheet

This Thesis entitled Continuous Time Identification of Systems with known and unknown order by Varanasi Santhosh Kumar is approved for the degree of Master of Technology from IIT Hyderabad

(Dr. Ketan P. Detroja) Examiner
Dept. of Electrical Engineering
IITH

Kishalay Mitra

(Dr. Kishalay Mitra) Examiner
Dept. Chemical Engineering
IITH

(Dr. Phanindra V. Jampana) Adviser
Dept. of Chemical Engineering
IITH

(Dr. C. S. Sastry) Chairman
Dept. of Mathematics
IITH

Acknowledgements

Firstly, I would like to thank my parents for their support and my teacher Mr. Gopala Krishna garu for inspiring me to do higher studies. I would like to express my sincere gratitude to my adviser Dr. Phanindra Jampana for his immense patience, support,motivation and guidance in doing this work. I would like to thank my Thesis Evaluation committee Dr. Ketan P. Detroja, Dr. Challa Subrahmanya Sastry and Dr. Kishalay Mitra for their patient listening and reviewing in the presentations and giving some valuable suggestions. I am also thankful to the Institute for providing the Teaching Assistantship during two years.

Dedication

To my Parents, Teachers and Friends.

Abstract

System Modelling is important in the field of process control. In this thesis, we reviewed the different kinds of modelling strategies and explained the importance of Continuous time models. Some of the early methods such as LSSVF and RIVC and new methods such as PDA and GS were reviewed. In case of noiseless systems model order is estimated using zero-norm minimization whereas for noisy systems order estimation is done using a brute force search.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Motivation

In the field of process control it is essential to identify the model, before designing a suitable control strategy. The most general way of building a model is based on the first or the fundamental principles i.e., on the basis of the physics of system. The main problem with such kind of approach is that the model may become complex and cannot be used in design of control strategy.

An alternative approach to physics based modelling is data-based identification, which can be classified broadly into three categories.

- **Grey Box Models** , are the models that are constructed from the basic physical principles and the parameters that represent the system coefficients, atleast have a direct physical interpretation.

- **Black Box Models** , are flexible models that can be used in general applications, in which the parameters may or may not have any physical interpretation.

- **Data Based Mechanistic Models** , are similar to that of the black box models but are meaningful only when the identified parameters have some physical interpretation.

If system obeys the zero-order hold (ZOH) assumption, then discrete time identification techniques are equivalent to the continuous time identification techniques. Modelling the system without considering the ZOH assumption is the main motivation for doing Continuous Time Identification. One more important aspect is that, as most of the physical systems are continuous in nature, a continuous time model may provide physical understanding of the system. If a system has both fast and slow dynamics, one cannot build a discrete time model for such cases [1] and CTI is the best alternative.

1.2 Definitions

1.2.1 System Identification:

Any System Identification problem is characterized by the following

- a class of models
- a class of input-output signals
- a criterion

System identification methodologies can be broadly classified into two categories

1. Discrete Time Identification
2. Continuous Time Identification

1.2.2 Zero Order Hold:

When the input between two consequent time instances remains constant, then it is called a ZOH system whose mathematical representation is $u(t) = k$ where, $T_i < t < T_{i+1}$

1.2.3 Linear Time Invariant System

Any Linear Time Invariant System (LTI) system can be represented mathematically by

- An n^{th} order differential equation, can be represented as

$$a_n \frac{d^n y}{dt^n} + a_{n-1} \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_o y = b_m \frac{d^m u}{dt^m} + b_{m-1} \frac{d^{m-1} u}{dt^{m-1}} + \ldots + b_o u \tag{1.1}$$

- A transfer function
- n-first order differential equations
- A matrix differential equation i.e., state space model

A state space representation is a mathematical model of representing the physical system as a set of input, output and state variables which are related by a system of first order differential equations.

- State Variable: A state variable of a system is one of the set of variables that describes enough about the system to determine its future behaviour in the absence of any external forces affecting the system.

In general, any LTI system can be described using state variables as

$$\begin{aligned}
\tfrac{dx_1}{dt} &= f[x_1, x_2, x_3, \ldots, x_n, u_1, u_2, \ldots, u_n] \\
\tfrac{dx_2}{dt} &= f[x_1, x_2, x_3, \ldots, x_n, u_1, u_2, \ldots, u_n] \\
\tfrac{dx_3}{dt} &= f[x_1, x_2, x_3, \ldots, x_n, u_1, u_2, \ldots, u_n] \\
&\vdots \\
\tfrac{dx_n}{dt} &= f[x_1, x_2, x_3, \ldots, x_n, u_1, u_2, \ldots, u_n]
\end{aligned}$$

Representing all these equations in the form of a matrix, gives a state space model of generalized form as

$$\tfrac{dX}{dt} = AX$$

Thus, the most general state space representation of a linear system with "p" inputs , "q" outputs and "n" state variables is

$$\begin{aligned}
\frac{dX(t)}{dt} &= A(t)X(t) + B(t)U(t) \\
Y(t) &= C(t)X(t) + D(t)U(t)
\end{aligned}$$

Where

$X(t)$: State Vector, of dimension $n \times 1$

$U(t)$: Input Vector, of dimension $p \times 1$

$Y(t)$: Output Vector, of dimension $q \times 1$

$A(t)$: State or System matrix, of dimension $n \times n$

$B(t)$: Input matrix, of dimension $n \times p$

$C(t)$: Output matrix, of dimension $q \times n$

$D(t)$: feed through matrix, of dimension $q \times p$

Chapter 2

LITERATURE REVIEW

Continuous Time Identification (CTI) has gained much of its current importance from the year 1960, prior to which, dynamic systems modelling was done using either frequency or transient response data. The rapid development in the methods of parameter estimation in the domain of discrete-time modelling, allowed the scientific community to draw parallels between it and its counterpart, continuous time modelling, leading to simultaneous development in the domain of CTI [2].

Consider the following generic representation of a system of ordinary differential equation (ODE),

$$\frac{dX}{dt} = f(x, u, t/\theta) \tag{2.1}$$

In general, in case where the analytical solution of the system of ODEs mentioned above is not available, the method of Laplace transform might be used to estimate the parameters of the system of linear ODEs when they are time independent. While in most of the other cases, several numerical methods may be applied, such as least square regression, etc. to estimate the parameters. The dependency of the solution of system of ODEs on the Laplace variable imposes certain restrictions on the selection of inputs into the system to ensure stability of the algorithm [3]. While the numerical methods to solve the system of ODEs, would either fail with increase in number of the parameters or again suffer with the issues of instability.

A significant work [4] reported in the literature suggested two kind of approaches viz.

a. To identify a discrete time model from the given data and then to effectively convert the discrete time model into the continuous time model. Although there exists considerable amount of work in literature to identify the discrete time models, conversion from discrete time to continuous time domain is not accurate [4].

b. To directly identify the continuous time model from the sampled data, where one can proceed with integrating the obtained system of ODEs. However, a major flaw in this formulation lies with the fact that the estimation of integrals is not trivial in all cases, which sometimes might lead to a highly intense and laborious exercise.

Garnier et. al. used Equation Error(EE) methods to fit a continuous -time transfer function model to the discrete time data, and evaluated the performance of the models by using Monte-Carlo Simulations [1]. The experiments have been carried out to determine the parameters, sampling period, type of input signal. The general scheme for EE model structure-based estimation of a continuous time model from a discrete time measurement requires two stages

- Primary stage which consists of finding a pre-processing method to generate some measures of process signals and their time derivatives. Pre-processing methods are broadly classified into three categories

  - Linear filters
  - Integral Methods
  - Modulating functions

  This stage also includes finding a discretizing technique so that the pre-processing operation can be performed from the sampled input-output data.

- Secondary stage in which CT parameters are estimated within the frame work of estimation method.

Ramsay et. al. [5] suggested a two step optimization based procedure to estimate the parameters where, the first step being fitting the observed data using certain basis functions to enable the evaluation of differentials necessary in the subsequent step. While, the second step solves the obtained system of higher order differentials in order to estimate the parameters.

Chapter 3

Laplace Transform based Identification

3.1 Problem Statement

The problem of system identification can be stated as the estimation of the parameters "A", "B", "C" and "D" of state space model or the parameters $a_n, a_{n-1}, \dots, a_o, b_m, b_{m-1}, \dots, b_o$ of Linear Time Invariant (LTI) system from the input-output data: $U(t) - Y(t)$. For the sake of simplicity, we have assumed that the parameters A, B, C are time independent while D is a zero vector. We focused on a single input single output (SISO) formulation which, without loss of any generality, can be easily extended to a multiple input multiple output (MIMO) systems.

3.2 Proposed Method

The proposed method is based on the fact that, for the systems where the aforementioned assumptions hold good and initial values of the states are zeros, the Laplace transformed equation results in a system of linear equations which in most cases can be solved easily. Thus for such systems, the state space equations can now be written as follows:

$$\frac{dX(t)}{dt} = AX(t) + BU(t) \tag{3.1}$$

$$Y(t) = CX(t) \tag{3.2}$$

Now, applying Laplace transform on both sides results in the following equation:

$$\begin{aligned} sX(s) - X(0) &= AX(s) + BU(s) \\ Y(s) &= cX(s) \end{aligned}$$

As $X(0) = 0$, the final form of transformed equations are

$$\begin{aligned} sX(s) &= AX(s) + BU(s) \\ Y(s) &= CX(s) \end{aligned}$$

Modifying the above equation gives,

$$X(s) = [sI - A]^{-1} BU(s) \tag{3.3}$$

Thus Final $Y(s)$ can be written as,

$$Y(s) = C[sI - A]^{-1} BU(s) \tag{3.4}$$

Truncating the expansion $[sI - A]^{-1}$ to the first $n + 1$ terms, we have

$$\Rightarrow Y(s) = -[CA^{-1}B + CA^{-2}B + CA^{-3}B + CA^{-4}B + \cdots + CA^{-(n+l)}B]U(s)$$

$$\Rightarrow Y(s) = -\begin{bmatrix} CA^{-1}B & CA^{-2}B & CA^{-3}B & \ldots & CA^{-(n+l)}B \end{bmatrix} \begin{bmatrix} 1 \\ s \\ s^2 \\ s^3 \\ \ldots \\ s^n \end{bmatrix} U(s)$$

The above equation is of the form $Y = -G * W$ where

$$\begin{aligned} Y &= Y(s) \\ G &= \begin{bmatrix} CA^{-1}B & CA^{-2}B & CA^{-3}B & \ldots & CA^{-(n+l)}B \end{bmatrix} \\ W &= \begin{bmatrix} 1 & s & s^2 & s^3 & \ldots & s^n \end{bmatrix}^T U(s) \end{aligned}$$

Although the above expressions are derived only for a single value of s, it can be shown that a similar derivation can be obtained for multiple values of s. In such cases where one encounters multiple values of s, Y will be a vector, W will be decomposed into two matrices in which one is a pure s matrix, known as Vandermonde matrix while the other one is the "input matrix".

For example, considering n values of s, then the system of equations retains the same structure of $\mathbf{Y} = \mathbf{G}\times\mathbf{W}$, but

$$Y = \begin{bmatrix} Y(s_1) & Y(s_2) & Y(s_3) & \dots & Y(s_N) \end{bmatrix}$$

$$G = \begin{bmatrix} CA^{-1}B & CA^{-2}B & CA^{-3}B & \dots & CA^{-(n+l)}B \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ s_1 & s_2 & s_3 & \dots & s_N \\ s_1^2 & s_2^2 & s_3^2 & \dots & s_N^2 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ s_1^n & s_2^n & s_3^n & \dots & s_N^n \end{bmatrix} \begin{bmatrix} U(s_1) & 0 & 0 & \dots & 0 \\ 0 & U(s_2) & 0 & \dots & 0 \\ 0 & 0 & U(s_3) & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & U(s_N) \end{bmatrix}$$

The solution to the equation $\mathbf{Y} = \mathbf{G}\times\mathbf{W}$ is given below:

$$G = (Y * W^T)^{-1}(W * W^T) \tag{3.5}$$

Here since the direct estimation of Laplace transform is impossible, the Laplace transforms are calculated numerically by using the equation

$$U(s) = L\{U(t)\} = \int_0^\infty e^{-st}U(t)dt = \delta \sum_{k=0}^{T} e^{-sk\delta}U(k\delta) \tag{3.6}$$

The major issues regarding the solution of the equation is

- The matrix W should be of full row rank.

- Numerical estimation of the Laplace transform and selection of values of s, as bad selection leads to the matrix W to become highly ill conditioned.

One possible solution to the aforementioned problems might be the selection of values of s lying on a unit circle which would lead to W being the Discrete Fourier Transform Matrix, whose condition number is one thus ensuring stability of the system. The general formula for selecting s values are

$$s_k = e^{-\frac{i2k\pi}{n+1}}, k = 0, 1, \ldots, n \tag{3.7}$$

But as mentioned previously the choice of s restricts the selection of input to that whose poles lie in a region with real part less than -1, an example to such an input is exponentially decaying function. One more restriction is that the number of s values that are needed to be chosen should be equal to the series truncating number i.e., $N = n + 1$

3.3 Numerical Experimentation:

For a simple illustration consider a simple single tank system which is a simple first order system as shown in the given figure.

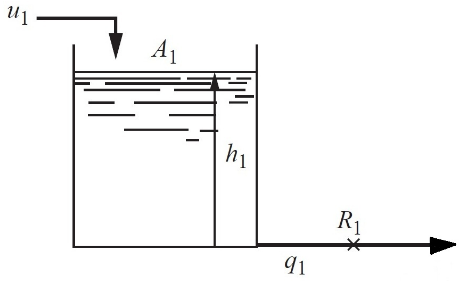


Figure 3.1: Simple Tank System

Writing a simple mass balance to such a system,

$$A\frac{dh_1}{dt} = u_1 - q_1 \tag{3.8}$$

where

$$q_1 = \frac{h_1}{R_1}$$
$$\Rightarrow \frac{dh_1}{dt} = \frac{u_1}{A_1} - \frac{h_1}{A_1 R_1}$$

After rearranging,

$$\Rightarrow \frac{dh_1}{dt} = -\frac{1}{A_1 R_1} h_1 + \frac{u_1}{A_1} \tag{3.9}$$

$$Y = h_1 \tag{3.10}$$

Comparing the above equations with the state apace equation, the height of the tank was the state variable and the parameters are:

$$a = -\frac{1}{A_1 R_1}$$
$$b = \frac{1}{A_1}$$
$$c = 1;$$
$$d = 0;$$

3.4 Results and Conclusions

Consider a state space model with exponentially decaying function as input and with the variables $\mathbf{a = -2, b = 1, c = 1}$ and $\mathbf{d = 0}$ for obtaining the input-output data. Once the input-output data were obtained, the proposed method was utilized to solve the problem for obtaining the values of a,b,c and d. The results of the proposed method are compared with the original values in Table 3.1. Instead of individual values of a,b,c and d, one can obtain a the values in groups such as cb/a ... etc. using which, one can exactly estimate only single parameter "a" and if in the remaining parameters if one parameter is fixed the other one can be estimated.

Table 3.1: Comparision of actual method and the proposed method for a first order system

	proposed method	original value
$CA^{-1}B$	2.000274016	2
$CA^{-2}B$	2.00042701	2
$CA^{-3}B$	2.000590941	2
$CA^{-4}B$	2.000899333	2
$CA^{-5}B$	2.001616662	2
$CA^{-6}B$	2.003224625	2
$CA^{-7}B$	2.006511801	2

Consider a case of higher order system of the state space form with

$$A = \begin{bmatrix} 0 & 1 \\ -6 & -5 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Again following the above algorithm , the results are tabulated below

Table 3.2: Comparison of actual method and the proposed method for second order system.

	proposed method	original value
$CA^{-1}B$	0.184336	0.166667
$CA^{-2}B$	0.144282	0.138889
$CA^{-3}B$	0.089778	0.087963
$CA^{-4}B$	0.050743	0.050154
$CA^{-5}B$	0.02731	0.027135
$CA^{-6}B$	0.014289	0.014253
$CA^{-7}B$	0.007339	0.007355
$CA^{-8}B$	0.003715	0.003754
$CA^{-9}B$	0.001852	0.001902

Thus, from the above tables, this can be concluded that the proposed method works good for single input single output (SISO) case as well as for a higher order systems.

Chapter 4

Parameter estimation of Linear Time Invariant System

4.1 Introduction

Consider a system which takes an input vector $u(t)$ and does certain operations and transforms it into the output vector $y(t)$. Such systems can be represented by the following form of ODE.

$$\frac{dy}{dt} = f(y, u, t/\theta) \tag{4.1}$$

where the vector, θ contains all the parameters defining the function f, which needs to be estimated. The function f, can either be linear or non-linear. In this chapter, we has assumed the function f to be always linear. The general form of an equation which represents the LTI system is described below:

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1}y}{dt^{n-1}} + \ldots + a_o y = b_m\frac{d^m u}{dt^m} + b_{m-1}\frac{d^{m-1}u}{dt^{m-1}} + \ldots + b_o u \tag{4.2}$$

where

$$\theta = \begin{bmatrix} a_{n-1} & a_{n-2} & \ldots & a_o & b_m & b_{m-1} & b_{m-2} & \ldots & b_o \end{bmatrix} \tag{4.3}$$

are the parameters vector that needs to be estimated.

Here, one can use the Laplace transform tool to estimate the parameters which on one hand facilitates the individual estimation of parameters [3]. While on the other hand this method of Laplace transforms suffers with certain specific problems as disscussed in the previous chapter.

This chapter is organized as follows. The subsequent section describes the algorithms which are well established in literature such as Least Squares based State Variable Filter (LSSVF) [6] and Refined Instrumental variables for Continuous Time systems (RIVC) [7] along with some recently developed algorithms such as Principal Differential Analysis (PDA) [8], Generalized Smoothing splines (GS) [5]. The following section presents a comprehensive comparison study of all the aforementioned methods in both noise as well as noise-less systems along with some numerical simulations. The final section presents the algorithm to estimate the parameters for the systems with unknown order.
The following three steps remain unchanged, irrespective of the algorithm.

- Define a predictor, $\hat{y}(t, \theta)$
- Form an error function, $e(t) = y(t) - \hat{y}(t, \theta)$ and pass it through a filter, F to filter the noise.
- Form an optimization problem with this filtered errors.

Depending upon the way of filtering and the formation of optimization problem, the algorithms differ from each other.

4.2 Least Squares based State Variable Filter

Let $x(t)$ denotes the noise-free output. In such case, the equation (4.2) can be written as

$$\frac{d^n x}{dt^n} + a_{n-1}\frac{d^{n-1}x}{dt^{n-1}} + \ldots + a_o x = b_m \frac{d^m u}{dt^m} + b_{m-1}\frac{d^{m-1}u}{dt^{m-1}} + \ldots + b_o u \tag{4.4}$$

The above equation can be further simplified to

$$A(p)x(t) = B(p)u(t) \tag{4.5}$$

where,

$$\begin{aligned} A(p) &= p^n + p^{n-1} + \cdots + p^0 \\ B(p) &= p^m + p^{m-1} + \cdots + p^0 \\ p^i &= \frac{d^i}{dt^i} \end{aligned}$$

The filter used in LSSVF is of the form

$$F(p) = \frac{1}{(p+\lambda)^k} \tag{4.6}$$

where, k should be greater than or equal to n. For sake of simplicity we considered $k = n$. Thus, the equation 4.4 can be expressed as

$$x_f^{(n)}(t_k) = \Phi^T(t_k)\Theta^T \tag{4.7}$$

where,

$$x_f^i(t) = f_i(p)x(t)$$
$$u_f^i(t) = f_i(p)u(t)$$

and $f_i(p) = \frac{p^i}{(p+\lambda)^n}$

Thus, the solution vector is of the form

$$\hat{\theta} = \Big(\sum_{k=1}^{n} \Phi(t_k)\Phi^T(t_k)\Big)^{-1}\Big(\sum_{k=1}^{n} \Phi(t_k)x_f^n(t_k)\Big) \tag{4.8}$$

where,

$\Phi^T(t_k) = \begin{bmatrix} -x_f^{(n-1)}(t_k) & -x_f^{(n-2)}(t_k) & \dots & -x_f^{(0)}(t_k) & u_f^{(m)}(t_k) & u_f^{(m-1)}(t_k) & \dots & u_f^{(0)}(t_k) \end{bmatrix}$

and $\theta = \begin{bmatrix} a_{n-1} & a_{n-2} & \dots & a_o & b_m & b_{m-1} & b_{m-2} & \dots & b_o \end{bmatrix}$

In case of an additive noise, the output can be expressed as $y(t_k) = x(t_k) + \eta(t_k)$ and the solution remains same with change in Φ vector as depicted below

$\Phi^T(t_k) = \begin{bmatrix} -y_f^{(n-1)}(t_k) & -y_f^{(n-2)}(t_k) & \dots & -y_f^{(0)}(t_k) & u_f^{(m)}(t_k) & u_f^{(m-1)}(t_k) & \dots & u_f^{(0)}(t_k) \end{bmatrix}$

4.3 Refined Instrumental Variable Method for Continuous Time Models

Here the error function is defined as $\epsilon(t_k) = y(t_k) - \frac{B(p)}{A(p)}u(t_k)$, which can be modified as $\epsilon(t_k) = A(p)y_f(t_k) - B(p)u_f(t_k)$ with $y_f(t_k) = \frac{1}{A(p)}y(t_k)$ and $u_f(t_k) = \frac{1}{A(p)}u(t_k)$ and unlike LSSVF method, here the filtering mechanism is dependent on the differential equation thus making it as an iterative procedure.

The RIVC algorithm is a two stage algorithm where the first stage is the initialization stage while the second one is the actual procedure wherein each stage consists of four to five steps.

In the initialization step, to estimate the initial parameters, SRIVC algorithm was used which is same as RIVC but with slight modifications which are explained below.

Stage 1: Initialization step

Step 1: Design a stable filter similar to the one in LSSVF method, i.e., $F(p) = \frac{1}{(p+\lambda)^n}$

Step 2: Filter the input-output data with this filter and also estimate the derivatives of the filtered signals.

Step 3: Thus, the error function is $x_f^{(n)}(t_k) = \Phi^T(t_k)\Theta^T$ and the solution is

$$\hat{\theta} = \Big(\sum_{k=1}^{n} \Phi(t_k)\Phi^T(t_k)\Big)^{-1}\Big(\sum_{k=1}^{n} \Phi(t_k)x_f^n(t_k)\Big) \tag{4.9}$$

where,

$$\Phi^T(t_k) = \Big[-y_f^{(n-1)}(t_k) \quad -y_f^{(n-2)}(t_k) \quad \dots \quad -y_f^{(0)}(t_k) \quad u_f^{(m)}(t_k) \quad u_f^{(m-1)}(t_k) \quad \dots \quad u_f^{(0)}(t_k)\Big]$$

where,

$$\begin{aligned} y_f^i(t) &= f_i(p)y(t) \\ u_f^i(t) &= f_i(p)u(t) \\ f_i(p) &= \tfrac{p^i}{(p+\lambda)^n} \end{aligned}$$

Step 4: Proceed with the algorithm developed in stage 2 (as described below) but without the discrete filter, known as the SRIVC method.

After estimating the initial parameters in stage-1, the stage-2 procedure is as follows

Stage 2: Actual procedure, for j=1:convergence

Step 1: Estimate $\hat{x}(t_k) = \frac{\hat{B}_j(p)}{\hat{A}_j(p)}u(t_k)$

Step 2: Estimate the error $\eta(t_k) = y(t_k) - \hat{x}(t_k)$ and use least squares to solve

$$\text{error function} = \eta(t_k) - \frac{1}{D_j(q^{-1})}e(t_k)$$

where, $e(t_k)$ is a white noise.

Step 3: Filter $\hat{x}(t_k), y(t_k), u(t_k)$ and also their derivatives by passing it into a continuous filter, $\frac{1}{\hat{A}_j(p)}$ and then through discrete filter, $D_j(q^{-1})$

Step 4: Thus the error function is $\epsilon(t_k) = \hat{\Phi}(t_k)(y_f^n(t_k) - \Phi^T(t_k)\hat{\theta}_j)$ and solution is of the form

$$\hat{\theta}_j = \Big(\sum_{k=1}^{n} \hat{\Phi}(t_k)\Phi^T(t_k)\Big)^{-1}\Big(\sum_{k=1}^{n} \Phi(\hat{t}_k)x_f^n(t_k)\Big) \tag{4.10}$$

Where,

$$\Phi^T(t_k) = \Big[-y_f^{(n-1)}(t_k) \quad -y_f^{(n-2)}(t_k) \quad \dots \quad -y_f^{(0)}(t_k) \quad u_f^{(m)}(t_k) \quad u_f^{(m-1)}(t_k) \quad \dots \quad u_f^{(0)}(t_k)\Big]$$

$$\hat{\Phi}^T(t_k) = \Big[-x_f^{(n-1)}(t_k) \quad -x_f^{(n-2)}(t_k) \quad \dots \quad -x_f^{(0)}(t_k) \quad u_f^{(m)}(t_k) \quad u_f^{(m-1)}(t_k) \quad \dots \quad u_f^{(0)}(t_k)\Big]$$

where,

$$\begin{aligned} y_f^i(t_k) &= \tfrac{D_j(q^{-1})}{\hat{A}_j(p)}p^i y(t_k) \\ u_f^i(t_k) &= \tfrac{D_j(q^{-1})}{\hat{A}_j(p)}p^i u(t_k) \\ x_f^i(t_k) &= \tfrac{D_j(q^{-1})}{\hat{A}_j(p)}\tfrac{\hat{B}_j(p)}{\hat{A}_j(p)}p^i x(t_k) \end{aligned}$$

4.4 Principal Differential Analysis:

Similar to the LSSVF method this is not an iterative procedure and on an overview, this method consists of two steps

- Fitting the splines to the output data with or without the penalty function(for smoothing purpose)
- Solving the ode to obtain the parameters

First, we give some details about B-splines and their construction.

B-splines and its Construction

4.4.1 Construction

A B-spline is a piece wise polynomial of some order, fitted between points.

Let "U" be a set of "$m+1$" non-decreasing sequence of numbers of form u_i, called the knots and the P_i^s are the control points then the general n^{th} order spline is expressed as a recursive relation of the form

$$N_{i,p} = \frac{t - u_i}{u_{i+p} - u_i} N_{i,p-1} + \frac{u_{i+p+1} - t}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1} \tag{4.11}$$

where

$$N_{i,0} = \begin{cases} 1, & \text{if } u_i \leq t < u_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{4.12}$$

and the curve is defined by

$$C(t) = \sum_{i=0}^{N} P_i N_{i,p}(t) \tag{4.13}$$

The Derivative of the B-spline curve is given by the general expression

$$\frac{dN_{i,p}}{dt} = \frac{p}{u_{i+p} - u_i} N_{i,p-1} + \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1} \tag{4.14}$$

4.4.2 Methodology:

Here, the actual LTI system is modified with considering coefficient of the input term as one. The main reason for such a modification lies in the fact that, in such systems, the estimation of the order becomes more accurate. Thus, the modified LTI model is

$$a_n \frac{d^n y}{dt^n} + a_{n-1} \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_o y = b_m \frac{d^m u}{dt^m} + b_{m-1} \frac{d^{m-1} u}{dt^{m-1}} + \ldots + u \tag{4.15}$$

While solving the LTI system, the function formulation contains two parts, the first one gives information corresponding to how best one can fit the data to a particular equation and the second part corresponds to solving the ode.

Representing the things mathematically, in step-1, the equation to be solved was

$$H(c) = ||Y - \hat{Y}||^2 + \lambda \int_0^T \left(\frac{d^2 \hat{Y}}{dt^2} \right) \tag{4.16}$$

where

$$\hat{Y} = \sum_{k=0}^{N} c_k \phi_k$$

Differentiating the above equation w.r.t $c's$, gives

$$C = (\Phi^T \Phi - \Phi_d)^{-1}(\Phi Y) \tag{4.17}$$

where,

$$\Phi = \begin{bmatrix} \phi_1(t_1) & \phi_1(t_2) & \phi_1(t_3) & \dots & \phi_1(t_M) \\ \phi_2(t_1) & \phi_2(t_2) & \phi_2(t_3) & \dots & \phi_2(t_M) \\ \phi_3(t_1) & \phi_3(t_2) & \phi_3(t_3) & \dots & \phi_3(t_M) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_N(t_1) & \phi_N(t_2) & \phi_N(t_3) & \dots & \phi_N(t_M) \end{bmatrix} \quad \Phi_d = \begin{bmatrix} \int_0^T \phi_1^{11}\phi_1^{11}dt & \int_0^T \phi_1^{11}\phi_2^{11}dt & \dots & \int_0^T \phi_{11}^{1}\phi_N^{11}dt \\ \int_0^T \phi_2^{11}\phi_1^{11}dt & \int_0^T \phi_2^{11}\phi_2^{11}dt & \dots & \int_0^T \phi_2^{11}\phi_N^{11}dt \\ \int_0^T \phi_3^{11}\phi_1^{11}dt & \int_0^T \phi_3^{11}\phi_2^{11}dt & \dots & \int_0^T \phi_3^{11}\phi_N^{11}dt \\ \vdots & \vdots & \dots & \vdots \\ \int_0^T \phi_N^{11}\phi_1^{11}dt & \int_0^T \phi_N^{11}\phi_2^{11}dt & \dots & \int_0^T \phi_N^{11}\phi_N^{11}dt \end{bmatrix}$$

In the outer optimization step, the ODE is solved to obtain the parameter vector,

$$J(\theta) = \int_0^T [a_n \sum_{k=1}^{N} C_k \frac{d^n \phi_k}{dt^n} + \ldots + a_o \sum_{k=1}^{N} C_k \phi_k - (b_m \sum_{k=1}^{m} C_u \frac{d^m \phi_k}{dt^m} + \ldots + \sum_{k=1}^{m} C_u \phi_k)]^2 dt \tag{4.18}$$

The solution to such problem is found using any existing optimization formulations leading to:

$$\theta = (A)^{-1} B \tag{4.19}$$

where,

$$A = \begin{bmatrix} \int_0^T \frac{d^n\hat{Y}}{dt^n}\frac{d^n\hat{Y}}{dt^n}dt & \int_0^T \frac{d^{n-1}\hat{Y}}{dt^{n-1}}\frac{d^n\hat{Y}}{dt^n}dt & \dots & -\int_0^T \frac{d^m\hat{U}}{dt^m}\frac{d^n\hat{Y}}{dt^n}dt & \dots & -\int_0^T \frac{d\hat{U}}{dt}\frac{d^n\hat{Y}}{dt^n}dt \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ -\int_0^T \frac{d^n\hat{Y}}{dt^n}\frac{d^m\hat{U}}{dt^m}dt & -\int_0^T \frac{d^{n-1}\hat{Y}}{dt^{n-1}}\frac{d^m\hat{U}}{dt^m}dt & \dots & \int_0^T \frac{d^m\hat{U}}{dt^m}\frac{d^m\hat{U}}{dt^m}dt & \dots & \int_0^T \frac{d\hat{U}}{dt}\frac{d^m\hat{Y}}{dt^m}dt \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ -\int_0^T \frac{d^n\hat{Y}}{dt^n}\frac{d\hat{U}}{dt}dt & -\int_0^T \frac{d^{n-1}\hat{Y}}{dt^{n-1}}\frac{d\hat{U}}{dt}dt & \dots & \int_0^T \frac{d^m\hat{U}}{dt^m}\frac{d\hat{U}}{dt}dt & \dots & \int_0^T \frac{d\hat{U}}{dt}\frac{d\hat{U}}{dt}dt \end{bmatrix}$$

$$\theta = \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_0 \\ b_m \\ \vdots \\ b_1 \end{bmatrix} \quad B = \begin{bmatrix} \int_0^T \frac{d^n\hat{Y}}{dt^n}U(t)dt \\ \int_0^T \frac{d^n\hat{Y}}{dt^n}U(t)dt \\ \vdots \\ -\int_0^T \frac{d^n\hat{Y}}{dt^n}U(t)dt \\ \vdots \\ -\int_0^T \frac{d^n\hat{Y}}{dt^n}U(t)dt \end{bmatrix}$$

4.5 Generalized Smoothing Spline:

Similar to the RIVC, this method is also an iterative method. The main difference between the PDA method and this method is that the smoothing is based on the ODE which makes it more effective than PDA. It also has two steps wherein the first step corresponds to the data fitting and second step corresponds to the ode solving. The error function can be represented mathematically as follows:

$$J = J_1 + J_2 \tag{4.20}$$

where,

$$J_1 = ||Y - \sum_{k=1}^{N} C_k \phi_k||^2 \tag{4.21}$$

$$J_2 = \lambda \int_0^T [\dot{y} - f(y, u, t/\theta)]^2 dt \tag{4.22}$$

The objective function is of two kinds based on the input

- Known input

- Unknown input

If case of a known input, the problem is straight forward as the objective function is of the form

$$J = ||Y - \sum_{k=1}^{N} C_k \phi_k||^2 + \lambda \int_0^T [\sum_{k=1}^{N} C_k \frac{d^n \phi_k}{dt^n} + \ldots + a_o \sum_{k=1}^{N} C_k \phi_k - (b_m \frac{d^m u}{dt^m} + \ldots + b_o u)]^2 dt \tag{4.23}$$

Unlike the situation of known inputs, most of the problem formulations are with unkown inputs in which case the form of the input function is unknown. In such cases where the inputs are unknown but are devoid of noise, b-splines are fitted to the input data to estimate the general form of the input equation in terms of spline coefficients.

The objective function in such a case is of the form

$$J = ||Y - \sum_{k=1}^{N} C_k \phi_k||^2 + \lambda \int_0^T [\sum_{k=1}^{N} C_k \frac{d^n \phi_k}{dt^n} + \ldots + a_o \sum_{k=1}^{N} C_k \phi_k - (b_m \sum_{k=1}^{m} C_{uk} \frac{d^m \phi_k}{dt^m} + \ldots + b_o \sum_{k=1}^{m} C_{uk} \phi_k)]^2 dt \tag{4.24}$$

If the problem is such that the form of the input is known, then the optimization procedure is further divided into two steps.

- The Inner Optimization in which, the objective function is optimized with respect to the nuisance parameters alone keeping the structural parameters constant.

- The Outer Optimization in which the optimization is performed with respect to the structural parameters alone.

On the other hand, if the problem is of the form where the input form is unknown, apart from the two steps mentioned previously for the case of known inputs, an additional step has to be done to find the spline coefficients of input by minimizing the function

$$J_3 = ||U - \sum_{k=1}^{m} C_{uk}\phi_k||^2 \tag{4.25}$$

If the above function is optimized with respect to the coefficients, the solution vector of the form $C_U = [\Phi\Phi^T]^{-1}[\Phi U]$ will be obtained where,

$$C_U = [C_{u1}\ C_{u2}\ \ldots\ C_{um}]^T$$

The following sections are explained by taking the case of unknown input form into consideration.

4.5.1 Inner Optimization

During the Inner Optimization, the structural parameters are to be fixed and are differentiated with respect to the nuisance parameters c_i's. If the partial derivatives of the function with respect to the nuisance parameters are equated to zero, the following equation will be obtained:

$$[\lambda\Psi - \Phi\Phi^T]C = [\lambda Ua - \Phi Y] \tag{4.26}$$

and the solution for nuisance parameters will be

$$C = [\lambda\Psi - \Phi\Phi^T]^{-1}[\lambda Ua - \Phi Y] \tag{4.27}$$

where,

$$\Phi = \begin{bmatrix} \phi_1(t_1) & \phi_1(t_2) & \phi_1(t_3) & \dots & \phi_1(t_M) \\ \phi_2(t_1) & \phi_2(t_2) & \phi_2(t_3) & \dots & \phi_2(t_M) \\ \phi_3(t_1) & \phi_3(t_2) & \phi_3(t_3) & \dots & \phi_3(t_M) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_N(t_1) & \phi_N(t_2) & \phi_N(t_3) & \dots & \phi_N(t_M) \end{bmatrix} \quad C = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_N \end{bmatrix} \quad Y = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_M \end{bmatrix} \quad a = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}$$

$$U = \begin{bmatrix} \int_0^T \phi_1(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \int_0^T \phi_1^1(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \dots \\ \int_0^T \phi_2(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \int_0^T \phi_2^1(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \dots \\ \int_0^T \phi_3(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \int_0^T \phi_3^1(t)(b_m \sum_{k=1}^m C_u \frac{d^m u}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \dots \\ \vdots & \vdots & \dots \\ \int_0^T \phi_N(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \int_0^T \phi_N^1(t)(b_m \sum_{k=1}^m C_u \frac{d^m \phi_k}{dt^m} + \dots + b_o \sum_{k=1}^m C_u \phi_k)dt & \dots \end{bmatrix}$$

$$\Psi = \begin{bmatrix} \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_1^j \phi_1^p & \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_1^j \phi_2^p & \dots & \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_1^j \phi_N^p \\ \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_2^j \phi_1^p & \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_2^j \phi_2^p & \dots & \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_2^j \phi_N^p \\ \vdots & \vdots & \dots & \vdots \\ \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_N^j \phi_1^p & \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_N^j \phi_2^p & \dots & \sum_{p=0}^n \sum_{j=0}^n a_j a_p \int_0^T \phi_N^j \phi_N^p \end{bmatrix}$$

with a_n=1

The value of the nuisance parameters obtained from the inner optimization step will be utilized in outer optimization loop.

4.5.2 Outer Optimization

In order to perform the outer optimization which enables the estimation of the values of structural parameters θ, the function J_1 is considered which is an explicit function of nuisance parameters c_k, which are also explicit functions of θ. Assuming that the function J_1 is twice differentiable with respect to θ and c_k, the following equation will be obtained:

$$\frac{dJ_1}{d\theta} = \frac{\partial J_1}{\partial \theta} + \frac{\partial J_1}{\partial c}\frac{\partial c}{\partial \theta} \tag{4.28}$$

The dependence of nuisance parameters on θ is given by the Implicit Function Theorem which is explained in the further section.

4.5.2.1 Implicit Function Theorem

Statement of Theorem: Let $f : R^{n+m} \rightarrow R^m$ be a continuously differentiable function. The main aim is to construct a function $g : R^n \rightarrow R^m$ whose graph is precisely the set (x, y) such that $f(x, y) = 0$

Explanation:

As observed above, in order to find a map, all the solutions of the given function f are needed. Since it is not always possible to find all the solutions of f, a single point (a, b) is considered which is the solution of the above function "f" i.e., $f(a, b) = 0$ and a function "g" is constructed around that point (a, b) i.e., two open sets "U" and "V" are constructed around "a" and "b" respectively, such that the function $g : U \rightarrow V$ is the solution of "f" on "$U \times V$". Mathematically this can be represented as:

$$\{(x, g(x)) | x \in U\} = \{(x, y) \in U \times V | f(x, y) = 0\} \tag{4.29}$$

In order to state the implicit function theorem, the Jacobian matrix of f is defined at (a, b) as follows:

$$(Df)(a, b) = \left(\left| \begin{matrix} \frac{\partial f_1}{\partial x_1}(a, b) & \dots & \frac{\partial f_1}{\partial x_n}(a, b) \\ \vdots & \vdots & \vdots \\ \frac{\partial f_m}{\partial x_1}(a, b) & \dots & \frac{\partial f_m}{\partial x_n}(a, b) \end{matrix} \right| \left| \begin{matrix} \frac{\partial f_1}{\partial y_1}(a, b) & \dots & \frac{\partial f_1}{\partial y_m}(a, b) \\ \vdots & \vdots & \vdots \\ \frac{\partial f_m}{\partial y_1}(a, b) & \dots & \frac{\partial f_m}{\partial y_m}(a, b) \end{matrix} \right| \right) = [|X||Y|] \tag{4.30}$$

Then, from the statement of implicit function theorem, if Y is an invertible matrix then there always exists U,V and g and the following equation can be defined.

$$\frac{\partial g}{\partial x_j}(x) = -\left(\frac{\partial f}{\partial y}(x, g(x)) \right)^{-|} \left(\frac{\partial f}{\partial x_j}(x, g(x)) \right) \tag{4.31}$$

Applying Implicit function theorem to the outer optimization problem mentioned above results in

$$\frac{dc}{d\theta} = -\left(\frac{\partial^2 J}{\partial c^2} \right)^{-|} \left(\frac{\partial^2 J}{\partial c \partial \theta} \right) \tag{4.32}$$

and

$$\frac{dH}{d\theta} = -\frac{\partial H}{\partial c} \left(\frac{\partial^2 J}{\partial c^2} \right)^{-|} \left(\frac{\partial^2 J}{\partial c \partial \theta} \right) \tag{4.33}$$

which is to be estimated at $(\theta_o, c(\theta_o))$

Gauss Newton algorithm is needed to estimate $d^2H/d\theta^2$, which is given by the following equation:

$$\frac{d^2 H}{d\theta^2} = \frac{d}{d\theta} \left(\frac{dH}{d\theta} \right) \tag{4.34}$$

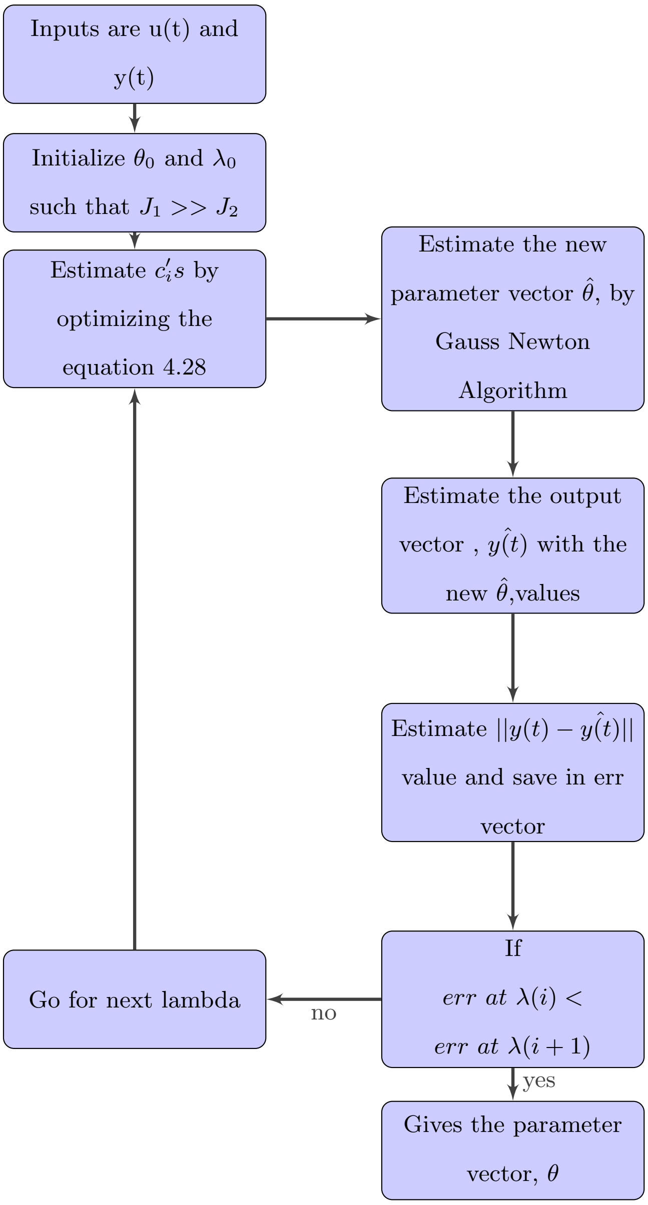
Substituting $dH/d\theta$ into the above equation, the final equation will be of the form,

$$\frac{d^2H}{d\theta^2} = -\left(\frac{\partial^2 J}{\partial c^2}\right)^{-1}\left(\frac{\partial^3 J}{\partial c\partial\theta^2}\right) + \left(\frac{\partial^2 J}{\partial c^2}\right)^{-1}\left(\frac{\partial^3 J}{\partial c^2\partial\theta}\right)\left(\frac{\partial^2 J}{\partial c^2}\right)^{-1}\left(\frac{\partial^2 J}{\partial c\partial\theta}\right) \tag{4.35}$$

Gauss Newton algorithm is used for estimation of the new $\theta's$ and the general step of Gauss Newton algorithm is

$$\theta_{n+1} = \theta_n - \gamma\left(\frac{d^2H}{d\theta^2}\right)^{-1}\frac{dH}{d\theta} \tag{4.36}$$

Upon obtaining the new $\theta's$, inner optimization step is solved to get the new $c's$ and the process is repeated till the convergence criteria is satisfied and the algorithm explained below.


Inputs are u(t) and y(t)
Initialize θ_0 and λ_0 such that $J_1 >> J_2$
Estimate $c_i's$ by optimizing the equation 4.28
Estimate the new parameter vector $\hat{\theta}$, by Gauss Newton Algorithm
Estimate the output vector , $y(\hat{t})$ with the new $\hat{\theta}$,values
Estimate $||y(t) - y(\hat{t})||$ value and save in err vector
If $err\ at\ \lambda(i) <$ $err\ at\ \lambda(i+1)$
no
Go for next lambda
yes
Gives the parameter vector, θ

4.6 Numerical simulations in Noise-less cases:

In all the following cases input u(t) is linear combination of sine wave of frequencies from 0.1 to 2.0 with a step of 0.1.

4.6.1 System 1:

$$\frac{dy}{dt} + y = u(t)$$

Table 4.1: Comparision of all Methods for noiseless first order system

Actual	LSSVF	RIVC	PDA		GS method
			lambda=0	lambda=0.1	
1	1.025	1.055	0.9995	0.7154	1.015
1	1.019	1.043	1.0001	1.0178	1.018

4.6.2 System 2:

$$\frac{d^2y}{dt^2} + 3\frac{dy}{dt} + y = u(t)$$

Table 4.2: Comparision of all Methods for noiseless second order system

Actual	LSSVF	RIVC	PDA		GS method
			lambda=0	lambda=0.1	
1	1.011	1.013	0.9905	0.724	0.98
3	3.034	3.023	2.992	2.284	2.9389
1	1.009	1.011	1	1.018	1.051

4.6.3 System 3:

$$\frac{d^4y}{dt^4} + 10\frac{d^3y}{dt^3} + 35\frac{d^2y}{dt^2} + 42\frac{dy}{dt} + 24y = u(t)$$

Table 4.3: Comparision of all Methods for noiseless fourth order system

Actual	LSSVF	RIVC	PDA		GS method
			lambda=0	lambda=0.1	
1	1.045	1.039	0.751	3.639	0.82
10	10.34	10.32	9.534	9.836	10.23
35	36.44	36.23	34.397	40.298	34.33
42	43.79	43.55	41.214	41.32	40.24
24	25.06	24.91	23.868	25.428	22.23

From the above results, we can conclude that all the aforementioned methods works well in case of a noiseless system except the PDA method with a defined λ value. The main reason for this is because of the penalty function in PDA is independent of system (ODE) what we have considered.

4.7 Numerical simulations in Noisy cases:

In general, there exists two kinds of noise:

a) Noise that is caused by the same signal of different frequency and amplitude

b) Random noise.

We consider a combination of both.

The input is a sine wave of frequency ranging from 0.1 to 2 with a step size of 0.1 and amplitude of 1. Thus the noisy sine signal is of the frequency 5 and amplitude 0.1 and the mean and variance of the random noise considered in this formulation is zero and σ, respectively. The mean to variance ratio of noise free signal to the noisy signal is at a minimum of 5.

4.7.1 System 1:

$$\frac{dx}{dt} + x = u(t)$$

$$y(t) = x(t) + \epsilon(t)$$

Table 4.4: Comparision of all Methods for noisy first order system

Actual	LSSVF	RIVC	PDA		GS method
			lambda=0	lambda=0.1	
1	1.014	1.132	0.9431	0.722	0.9602
1	0.9888	1.09	1.032	1.081	0.9428

4.7.2 System 2:

$$\frac{d^2x}{dt^2} + 3\frac{dx}{dt} + x = u(t)$$

$$y(t) = x(t) + \epsilon(t)$$

Table 4.5: Comparision of all Methods for noisy second order system

Actual	LSSVF	RIVC	PDA		GS method
			lambda=0	lambda=0.1	
1	0.6282	0.603	0.1831	0.7508	1.223
3	1.762	1.538	2.924	2.409	3.006
1	0.6505	0.5965	0.7818	1.002	1.273

From the above results, we can clearly see that GS algorithm works well than the PDA. This is because of the penalty function what we have considered in GS is ODE, which is not the case with PDA.

4.8 Model Order Estimation:

The algorithm described in the previous section gives the parameters in case of a known order. In general, the order is also a parameter which needs to be estimated. This section deals about the case of unknown order and its estimation using zero norm minimization [9]. First the algorithm is explained in a general case and then application of algorithm in PDA case is described in detail.

Representing the order estimation problem mathematically, the objective is to find sparse approximation of θ vector that minimizes some $f(\theta)$ i.e., $\min_{||\theta||_0 < s} f(\theta)$, where $f(\theta)$ can be a linear function or a non-linear function

If the function $f(\theta)$ is linear, one can use compressed sensing (CS) algorithms to estimate the model order. In both the PDA and the GS algorithms, the function, $f(\theta)$ is non-linear, which makes it difficult to use the CS algorithms. This leads to development of a new algorithm called the Greedy Sparse Algorithm.

4.8.1 Notations, Assumptions and Definitions:

For a given vector, $x \in \mathbb{R}^n$, the support set of x is defined as

$$I_1(x) \equiv \{i : x_i \neq 0\}$$

and its compliment is given by

$$I_0(x) \equiv \{i : x_i = 0\}$$

The set of vectors with at most $s-$sparse is given by C_s and

$$C_s = \{x : ||x||_0 \leq s\}$$

One assumption is that the function f is lower bounded.

Definitions:

4.8.1.1 Basic Feasible Vector:

A vector, $x^* \in C_s$ is called a $basic\ feasible$ vector of our problem, if

1. when $||x^*||_0 < s, \bigtriangledown f(x^*) = 0$
2. when $||x^*||_0 = s, \bigtriangledown_i f(x^*) = 0 \forall i \in I_1(x^*)$

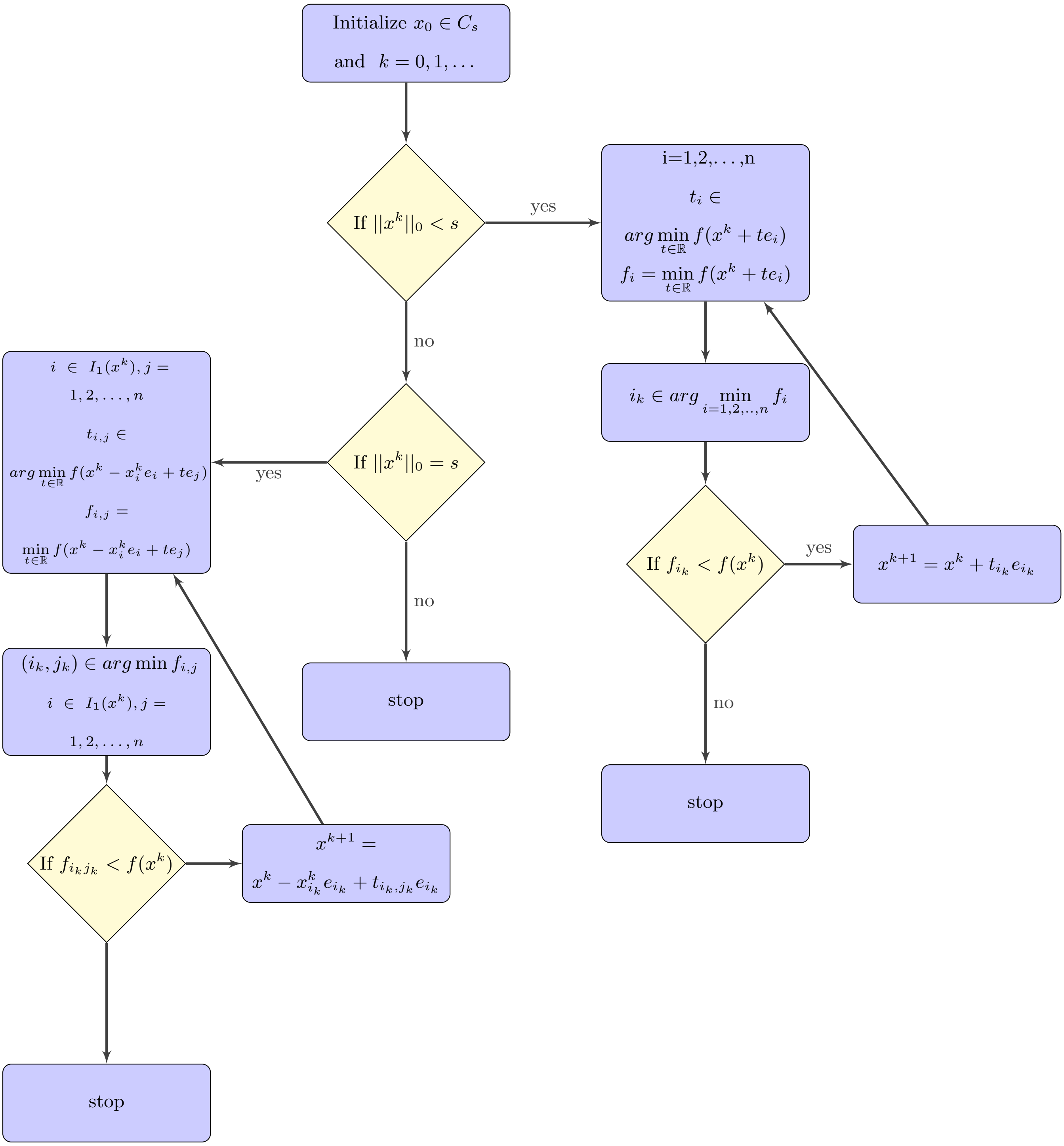
4.8.1.2 Coordinate-wise Minima:

Let x^* be a feasible solution of our problem ,then it is called a CW-Minima point if

1. $||x^*||_0 < s$ and $i = 1, 2, \ldots, n,\ f(x^*) = \min_{t \in \mathbb{R}} f(x^* + te_i)$
2. $||x^*||_0 = s$ and for every $i \in I_1(x^*)$ and $j = 1, 2, \ldots, n,\ f(x^*) \leq \min_{t \in \mathbb{R}} f(x^* - x_i^* e_i + te_j)$

In the hierarchy, any optimal point is a CW-Minima point and any CW-Minima point is a Basic Feasible vector (Proofs described in the subsequent chapter). So if one can have an algorithm that gives a CW-Minima point, we can get an approximate solution to our problem and the algorithm is explained below.

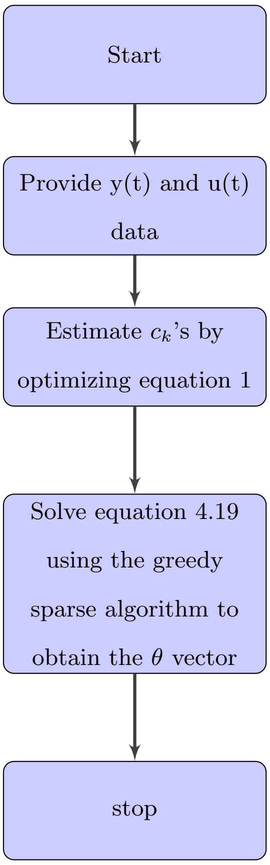
Greedy Sparse Algorithm


Initialize $x_0 \in C_s$
and $k = 0, 1, \ldots$
If $||x^k||_0 < s$
yes
i=1,2,...,n
$t_i \in$
$arg \min_{t\in\mathbb{R}} f(x^k + te_i)$
$f_i = \min_{t\in\mathbb{R}} f(x^k + te_i)$
no
$i_k \in arg \min_{i=1,2,..,n} f_i$
If $||x^k||_0 = s$
yes
$i \in I_1(x^k), j =$
$1, 2, \ldots, n$
$t_{i,j} \in$
$arg \min_{t\in\mathbb{R}} f(x^k - x_i^k e_i + te_j)$
$f_{i,j} =$
$\min_{t\in\mathbb{R}} f(x^k - x_i^k e_i + te_j)$
If $f_{i_k} < f(x^k)$
yes
$x^{k+1} = x^k + t_{i_k} e_{i_k}$
no
no
stop
$(i_k, j_k) \in arg \min f_{i,j}$
$i \in I_1(x^k), j =$
$1, 2, \ldots, n$
stop
If $f_{i_k j_k} < f(x^k)$
$x^{k+1} =$
$x^k - x_{i_k}^k e_{i_k} + t_{i_k, j_k} e_{i_k}$
stop

The main advantage with this algorithm is that CW-minima point is guaranteed (proved in subsequent chapter). If, we apply this algorithm to our PDA problem, the final algorithm will be as follows.

Equation 1:

$$H(c) = ||Y - \hat{Y}||^2 + \lambda \int_0^T \left(\frac{d^2\hat{Y}}{dt^2} \right)$$


Start
Provide y(t) and u(t) data
Estimate c_k's by optimizing equation 1
Solve equation 4.19 using the greedy sparse algorithm to obtain the θ vector
stop

This algorithm suffers with the disadvantages of the selection of sparsity level and initial values. In the subsequent paragraph we has proposed a method on the selection of the sparsity level and also the initial values. To start with we has assumed certain order which can be (say) g and let n be the actual order. 1. Solve PDA algorithm without the Sparse algorithm and obtain a dense solution of size $g + 1$ and let that solution be $\hat{\theta}_p$. Since this solution may contain negative values which may lead to an unstable system, these negative and near zero values are replaced with 0. Thus the dense solution now become sparse and the sparsity level was assumed to be the number of non-zero entries in the sparse vector. The application of this algorithm has provided the following results.

Noiseless case:

Two sub cases were considered in all the simulations owing to:

- Study the effect of variation of λ along with
- Study of order variation.

4.8.1.3 System 1:

$$2\frac{dy}{dt} + y = u(t)$$

Table 4.6: Model order estimation of First order system

lambda=0	lambda=0	lambda=0	lambda=0	lambda=0.1	lambda=0.1	lambda=0.1	lambda=0.1
order=2	order=3	order=4	order=5	order=2	order=3	order=4	order=5
0	0	0	0	0	0	-0.1033192	0
1.996544	0	0	0	1.74111067	0.12753425	0	0
0.999088	1.991579	0	0	1.03200426	1.28709489	0	0
	1.000091	1.992835	0		1.05456375	0	0
		1.00068	1.976941			1.15282621	2.22116543
			1.004941				1.03414023

4.8.1.4 System 2:

$$2\frac{d^2y}{dt^2} + 3\frac{dy}{dt} + y = u(t)$$

Table 4.7: Model order estimation of Second order system

lambda=0	lambda=0	lambda=0	lambda=0	lambda=0.1	lambda=0.1	lambda=0.1	lambda=0.1
order=3	order=4	order=5	order=6	order=3	order=4	order=5	order=6
0	0	0	0	0	0	0	0
1.953454	0	0	0	2.411117096	0	0	0
3.053112	1.987457	0	0	3.123129404	2.478826902	0	0
0.994333	3.050429	1.983596	0	1.049594347	3.132341843	2.484038096	0
	0.998954	3.082375	2.001301		1.058161359	3.096198824	2.519186804
		0.998337	3.077378			1.058984015	3.095405884
			1.001813				1.064746787

4.8.1.5 System 3:

$$2\frac{d^3y}{dt^3} + 3\frac{d^2y}{dt^2} + 3\frac{dy}{dt} + y = u(t)$$

Table 4.8: Model order estimation of Third order system

lambda=0	lambda=0	lambda=0	lambda=0.1	lambda=0.1	lambda=0.1
order=4	order=5	order=6	order=4	order=5	order=6
0	0	0	0	0	0
1.641471	0	0	1.805499099	1	-0.3947826
3.193595	1.698925	0	3.37620816	1	0
2.655207	3.043406	1.500747	2.783207934	5.06355616	0
1.03268	2.74524	3.283224	1.055673281	1	2.77043259
	1.007193	2.314762		1.17187804	1.78440873
		1.05454			0.97166016

From the above results, it can be concluded that in case of noiseless systems, with lambda=0, PDA algorithm works well. The reason for this might be the initial values of sparse step is itself a good approximation of the solution, which is not the situation in case of lambda=0.1.

4.9 Estimation using error plot:

The above algorithm works fine in case of noiseless systems but in presence of noise, the PDA does not work well and for such cases a Brute force method has been employed to estimate the model order.

In this procedure, we first fix the support of the parameter vector and use the GS-algorithm to obtain a solution. The errors for various sparsity levels are plotted and the sparsity at which the minimum occurs is considered to be the final solution.

4.9.1 Numerical simulations:

System1:

$$\frac{dx(t)}{dt} + x(t) = u(t)$$

$$y(t_k) = x(t_k) + \epsilon(t_k)$$

From the graph, at order=1 (which is the original order), the error is minimum and the solution was $a_0 = 0.9846,\ b_0 = 0.9226$

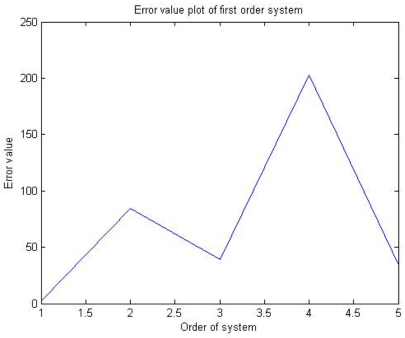


Figure 4.1: Error plot of first order System

System2:

$$\frac{d^2x(t)}{dt^2} + 3\frac{dx(t)}{dt} + x(t) = u(t)$$
$$y(t_k) = x(t_k) + \epsilon(t_k)$$

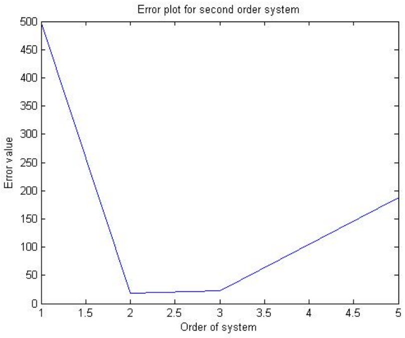


Figure 4.2: Error plot of second order System

From the graph, at order=2 and 3, the error is minimum, so either of these can be the order of system and for order=2, the solution was $a_1 = 2.239, a_0 = 0.8745, b_0 = 0.914$

System3:

$$\frac{d^3x}{dt^3} + 3\frac{d^2x}{dt^2} + 3\frac{dx}{dt} + x = u(t)$$
$$y(t_k) = x(t_k) + \epsilon(t_k)$$

From the graph, it is clear that at order=3, the error is minimum and the solution was $a_2 = 3.374, a_1 =$

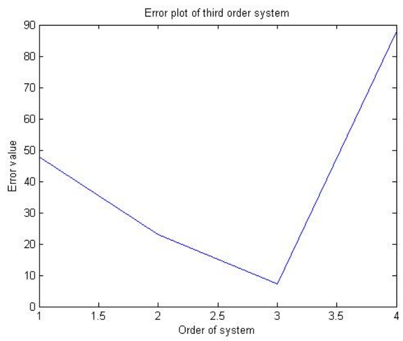


Figure 4.3: Error plot of third order System

$3.35, a_0 = 0.9719$ and $b_0 = 1.125$

4.10 Conclusions and Future Study:

- From the above results it can be concluded that PDA algorithm works well in the noise-less systems of both known and unknown orders. In the noisy systems the PDA failed in case of unknown order because of the arbitrary regularization used (i.e, the regularization is independent of the ODE fit).
- In GS algorithm, the smoothing is based on the ODE but the main problem with GS is to find out the initial parameters. So, either a hybrid method is needed which can perform well in all the cases or a GS algorithm is needed along with sparsity algorithm to estimate the order of the system.
- Apart from the spline basis, we will try to use wavelet basis and multi-resolution analysis, as wavelets are less susceptible to noise than splines.

Chapter 5

Convergence Analysis and Theorems

In this chapter, we briefly discuss about the convergence analysis of RIVC method as described in [10] and the proofs of the Greedy Sparse algorithm as explained in [9].

5.1 Convergence analysis of RIVC

Consider the RIVC method explained in the previous chapter and if the following assumptions are satisfied

- The true system $\frac{B^*(p)}{A^*(p)}$ is asymptotically stable and $A^*(p)$ and $B^*(p)$ are co-prime.
- $u(t_k)$ and $e(t_k)$ are stationary and mutually independent
- $u(t_k)$ is persistently exciting of order no less than $n + m + 1$
- All zeros of $\hat{A}_j(p)$ have negative real parts , $\hat{A}_j(p)$ and $\hat{B}_j(p)$ are co-prime and Real part of $\frac{A^*(p)}{\hat{A}_j(p)}$ is strictly positive.
- $\min(n - n^*, m - m^*) = 0$ where, (n, m) are orders of estimates and (n^*, m^*) are orders of true systems

then the following are true.

- The matrix $E\{\hat{\varphi}_f(t_k)\varphi_f^T(t_k)\}$ is non-singular.
- The true parameter θ^* is unique converging point.
- In the asymptotic case, the estimate $\hat{\theta}_{j+1}$ from RIVC method locally converges in one iteration to θ^* [10].

Proof. Before going to begin the proof , let us define a matrix called "Sylvester Matrix", it is a matrix that is associated with two uni-variate polynomials.

let P and Q are two polynomials defined as $P(Z) = p_0 + p_1 Z + p_2 Z^2 + \cdots + p_n Z^n$ and $Q(Z) = q_0 + q_1 Z + q_2 Z^2 + \cdots + q_m Z^m$, then Sylvester matrix is defined as

$$S_{P,Q} = \begin{bmatrix} p_m & p_{m-1} & \dots & p_o & 0 & \dots & 0 \\ 0 & p_m & p_{m-1}\dots & p_o & \dots & 0 & \\ \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ q_n & q_{n-1} & \dots & q_o & 0 & \dots & 0 \\ 0 & q_n & q_{n-1} & \dots & q_o & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \end{bmatrix}$$

The important property of this matrix is that if two polynomials do not have a common root then the matrix is always non-singular.

Revisiting some of the equations mentioned previously in chapter 4,

$$\hat{\varphi}_f(t_k) = \frac{\hat{D}_j(q)}{\hat{A}_j(p)} \begin{bmatrix} -\frac{\hat{B}_j(p)}{\hat{A}_j(p)} p^{n-1} u(t_k) & -\frac{\hat{B}_j(p)}{\hat{A}_j(p)} p^{n-2} u(t_k) & \dots & -\frac{\hat{B}_j(p)}{\hat{A}_j(p)} u(t_k) & p^m u(t_k) & p^{m-1} u(t_k) & \dots & u(t_k) \end{bmatrix}^T$$

$$\varphi_f(t_k) = \frac{\hat{D}_j(q)}{\hat{A}_j(p)} \begin{bmatrix} p^{n-1} y(t_k) & p^{n-2} y(t_k) & \dots & y(t_k) & p^m u(t_k) & p^{m-1} u(t_k) & \dots & u(t_k) \end{bmatrix}$$

Using $x(t) = \frac{B^*(p)}{A^*(p)} u(t)$ and $y(t_k) = x(t_k) + \epsilon(t_k)$ and if we modify the above equation then the final form of equation is

$$\varphi_f(t_k) = \frac{\hat{D}_j(q)}{\hat{A}_j(p)} \begin{bmatrix} p^{n-1}(B^*(p)u(t_k) + A^*(p)\epsilon(t_k)) \dots (B^*(p)u(t_k) + A^*(p)\epsilon(t_k)) & p^m A^*(p)u(t_k) \dots A^*(p)u(t_k) \end{bmatrix}$$

If we simplify $\hat{\varphi}_f(t_k)$ and $\varphi_f(t_k)$ using the Sylvester matrix then the final form of matrices are

$$\hat{\varphi}_f(t_k) = \frac{\hat{D}_j(q) S(-\hat{B}_j(p), \hat{A}_j(p))}{\hat{A}_j^2(p)} \begin{bmatrix} u^{n+m}(t_k) & u^{n+m-1}(t_k) & \dots & u(t_k) \end{bmatrix}^T$$

$$\varphi_f^T(t_k) = \frac{\hat{D}_j(q) S(-B_j^*(p), A_j^*(p))}{\hat{A}_j(p) A^*(p)} \begin{bmatrix} u^{n+m}(t_k) & u^{n+m-1}(t_k) & \dots & u(t_k) \end{bmatrix} - \frac{\hat{D}_j(q)}{\hat{A}_j(p)} \begin{bmatrix} p^{n+1}\epsilon(t_k) \dots \epsilon(t_k) & 0 \dots 0 \end{bmatrix}$$

So the first part is to prove that $E\{\hat{\varphi}_f(t_k)\varphi_f^T(t_k)\}$ is non-singular. Substituting $\hat{\varphi}_f(t_k)$ and $\varphi_f(t_k)$ in it gives

$$E\{\hat{\varphi}_f(t_k)\varphi_f^T(t_k)\} = S(-\hat{B}_j(p), \hat{A}_j(p)).\Phi.S(-B_j^*(p), A_j^*(p))$$

where

$$\Phi = E\Bigg\{ \frac{\hat{D}_j^2(q)}{\hat{A}_j^3(p) A^*(p)} \begin{bmatrix} u^{n+m}(t_k) & u^{n+m-1}(t_k) & \dots & u(t_k) \end{bmatrix}^T \begin{bmatrix} u^{n+m}(t_k) & u^{n+m-1}(t_k) & \dots & u(t_k) \end{bmatrix} \Bigg\}$$

and as from assumption-1 and assumption-4, it can be observed that both $S(-\hat{B}_j(p), \hat{A}_j(p))$ and $S(-B_j^*(p), A_j^*(p))$ are non-singular. Thus if we prove that the matrix Φ is non-singular then the proof of part-1 is completed.

Consider a stable filter of the form

$$f_{\bar{c}} = \frac{1}{(p+\lambda)^{n+m+1}}$$

$$u^{i}_{f_{\bar{c}}}(t) = f_{\bar{c}}(p)p^{i}u(t)\ for\ i = 0, 1, \dots, n+m$$

Thus

$$U_{f_{\bar{c}}}(t) = \begin{bmatrix} u^{n+m}_{f_{\bar{c}}}(t) & u^{n+m-1}_{f_{\bar{c}}}(t) & \dots & u_{f_{\bar{c}}}(t) \end{bmatrix}$$

If we express it in state-space form, then $\dot{U}_{f_{\bar{c}}} = A_{\bar{c}}U_{f_{\bar{c}}} + B_{\bar{c}}u(t)$ where

$$A_{\bar{c}} = \begin{bmatrix} -\alpha_1 & -\alpha_2 & \dots & -\alpha_{n+m+1} \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \dots & \vdots \\ 0 & \dots & 1 & 0 \end{bmatrix} \quad B_{\bar{c}} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where, $-\alpha_1, -\alpha_2, \dots, -\alpha_{n+m+1}$ are the coefficients of the polynomial $(p+\lambda)^{n+m+1}$ □

The discrete counterpart of the above state space-model is $\dot{U}_{f_{\bar{c}}}(t_k) = A_{\bar{d}}U_{f_{\bar{c}}(t_{k-1})} + B_{\bar{d}}u(t_{k-1})$ with $A_d = e^{A_{\bar{c}}h}$ and $B_{\bar{d}} = \int_0^h e^{A_{\bar{c}}\gamma}B_{\bar{c}}d\gamma$ where h is the difference between the sampling times, which can be simplified as

$$\begin{aligned} \dot{U}_{f_{\bar{c}}}(t_k) =& A_{\bar{d}}U_{f_{\bar{c}}}(t_{k-1}) + B_{\bar{d}}u(t_{k-1}) \\ &= A_{\bar{d}}(A_{\bar{d}}U_{f_{\bar{c}}}(t_{k-2}) + B_{\bar{d}}u(t_{k-2})) + B_{\bar{d}}u(t_{k-1}) \\ &= A^{k}_{\bar{d}}U_{f_{\bar{c}}}(t_0) + \Gamma U \end{aligned}$$

where $\Gamma = \begin{bmatrix} A^{k}_{\bar{d}}B_{\bar{d}} & A^{k-1}_{\bar{d}}B_{\bar{d}} & \dots & B_{\bar{d}} \end{bmatrix}$ and $U = \begin{bmatrix} u(t_1) & u(t_2) & u(t_3) & \dots & u(t_k) \end{bmatrix}^T$
As $f_{\bar{c}}(p)$ is stable it indicates that all the roots are negative which implies that the eigenvalue of the $A_{\bar{d}}$ lies inside the unit circle which implies that as $k \to \infty, A^{k}_{\bar{d}}U_{f_{\bar{c}}}(t_0) \to 0$ and the state-space is minimal which means that it has a minimum of $n+m+1$ states which implies the rank of the matrix Γ as $n+m+1$
Thus $E\{U_{f_{\bar{c}}}(t_k).U^{T}_{f_{\bar{c}}}(t_k)\} = \Gamma.E\{UU^T\}\Gamma^T$ and under assumption-3, the correlation matrix $E\{UU^T\}$ is positive definite which implies that $E\{U_{f_{\bar{c}}}(t_k).U^{T}_{f_{\bar{c}}}(t_k)\}$ is positive definite.
Now considering the matrix Φ and replacing $U_{f_{\bar{c}}}(t_k)$ in that, the final expression for Φ will be

$$\Phi = E\Bigg\{\Bigg(\frac{\hat{D}_j(q)}{\hat{A}_j(p)A^*(p)f_{\bar{c}}(p)}U_{f_{\bar{c}}}(t_k)\Bigg)\Bigg(\frac{\hat{D}_j(q)}{\hat{A}^2_j(p)f_{\bar{c}}(p)}{U_{f_{\bar{c}}}}^T(t_k)\Bigg)\Bigg\}$$

From assumption-4 along with the fact that $f_{\bar{c}}(p)$ is stable if one can substitute $s = j\omega$ in $\frac{\hat{D}_j(q)}{\hat{A}_j(p)A^*(p)f_{\bar{c}}(p)}$ then the real part of this expression is positive implying the positive definiteness of the Φ matrix and also its stability. The fact that Φ is non-singular implies that $E\{\hat{\varphi}_f(t_k)\varphi_f^T(t_k)\}$ is non-singular.

Now in the part-2, it is needed to prove that θ^* is the unique converging point.

Assume that $\bar{\theta}$ is limiting point of $\hat{\theta}_{j+1}$ and as $j \to \infty$, the equation

$$E\Big\{[\hat{\varphi}_f(t_k)(y_f^n(t_k) - \varphi_f^T(t_k)\hat{\theta}_{j+1})]|_{\hat{\theta}_{j+1}=\bar{\theta}}\Big\} = 0$$

From $x(t) = \frac{B(p)}{A(p)}u(t)$,$y(t_k) = x(t_k) + \epsilon(t_k)$ and from the definitions of $\hat{\varphi}_f(t_k)$,$\varphi_f^T(t_k)$, the above equation can be written as

$$\begin{aligned}
&E\Big\{[\hat{\varphi}_f(t_k)(y_f^n(t_k) - \varphi_f^T(t_k)\hat{\theta}_{j+1})]|_{\hat{\theta}_{j+1}=\bar{\theta}}\Big\} \\
&\qquad = E\left\{\frac{\bar{D}(q)S(-\bar{B}(p),\bar{A}(p)}{\bar{A}^2(p)}[u^{n+m}(t_k)\dots u(t_k)]^T.\Big(\bar{D}(q)\frac{\bar{A}(p)B^*(p) - \bar{B}(p)A^*(p)}{\bar{A}(p)A^*(p)}u(t_k) + \bar{D}(q)\epsilon(t_k)\Big)\right\} \\
&\qquad = E\left\{\frac{\bar{D}^2(q)S(-\bar{B}(p),\bar{A}(p)}{\bar{A}^3(p)A^*(p)}[u^{n+m}(t_k)\dots u(t_k)]^T[u^{n+m}(t_k)\dots u(t_k)]\right\}.H \\
&\qquad = 0
\end{aligned}$$

Where, $H = \bar{A}(p)B^*(p) - \bar{B}(p)A^*(p)$

It can be proved that the first term is non-singular and the proof is similar to that of the proof in part-1. As the expression is equated to zero it implies that $H = 0$ and from the assumptions-4 and assumption-5, only one solution to this problem can be obtained which is θ^*. This is because from assumption-5, the order of the system is clearly defined and from assumption-4, the solution should be positive definite. The consideration of both these assumptions reveals that the only possible solution is θ^*. Thus, θ^* is the unique converging point.

Thus the only part left is to prove that in asymptotic case, the solution converges locally in only one iteration. To prove this, certain variables have been redefined as follows, $\hat{\varphi}_f(t_k)$ as $\hat{\varphi}_f(t_k,\hat{\theta}_j)$ and $\varphi_f(t_k)$ as $\varphi_f(t_k,\hat{\theta}_j)$ and the following two new functions are introduced.

$$f_1(\hat{\theta}_j) := \Big(\frac{1}{N}\sum_{i=1}^{N}\hat{\varphi}_f(t_k,\hat{\theta}_j)\varphi_f^T(t_k,\hat{\theta}_j)\Big)$$

and

$$f_2(\hat{\theta}_j) := \Big(\frac{1}{N}\sum_{i=1}^{N}\hat{\varphi}_f(t_k,\hat{\theta}_j)\big(y_f^n(t_k) - \varphi_f^T(t_k,\hat{\theta}_j)\theta^*\big)\Big)$$

The linearisation of $\hat{\theta}_{j+1}$ around the limiting point $\bar{\theta}$ then from Taylor expansion truncated to first term gives

$$\hat{\theta}_{j+1} - \bar{\theta} \simeq f_1(\bar{\theta}) f_2(\bar{\theta}) + \left(\frac{\partial f_1(\hat{\theta}_j)}{\partial \hat{\theta}_j}\Big|_{\hat{\theta}_j=\bar{\theta}} f_2(\bar{\theta}) + f_1(\bar{\theta}) \frac{\partial f_2(\hat{\theta}_j)}{\partial \hat{\theta}_j}\Big|_{\hat{\theta}_j=\bar{\theta}} \right)(\hat{\theta}_j - \bar{\theta})$$

From proof of part-2, one can observe that as $N \to \infty$, $f_2(\bar{\theta}) = 0$

$$\begin{aligned} \Rightarrow\ \hat{\theta}_{j+1} - \bar{\theta} &\simeq f_1(\bar{\theta}) \frac{\partial f_2(\hat{\theta}_j)}{\partial \hat{\theta}_j}\Big|_{\hat{\theta}_j=\bar{\theta}} (\hat{\theta}_j - \bar{\theta}) \\ = \Big(\frac{1}{N}\sum_{i=1}^{N} \hat{\varphi}_f(t_k, \hat{\theta}_j)\varphi_f^T(t_k, \hat{\theta}_j)\Big) X \frac{1}{N}\sum_{i=1}^{N} \Big(&\frac{\partial \hat{\varphi}_f(t_k, \hat{\theta}_j}{\partial \hat{\theta}_j}\Bigg|_{\hat{\theta}_j=\bar{\theta}} \big(y_f^n(t_k) - \varphi_f^T(t_k, \bar{\theta}_j)\bar{\theta}\big)\Big) \\ &- \hat{\varphi}_f(t_k)\Big(\frac{\partial y_f^n(t_k)}{\partial \hat{\theta}_j}\Bigg|_{\hat{\theta}_j=\bar{\theta}} - \bar{\theta}^T \frac{\partial \varphi_f^T(t_k, \bar{\theta}_j)}{\partial \hat{\theta}_j}\Bigg|_{\hat{\theta}_j=\bar{\theta}} \Big) \end{aligned}$$

Thus, here if it can be proved that eigenvalue of matrix are stable or the terms are asymptotic independent, then $\hat{\theta}_{j+1}$ is locally convergent to $\bar{\theta}$ [10]
The first term in the RHS comprises of two terms and if it is expanded,

$$\begin{aligned} y_f^n(t_k) - \varphi_f^T(t_k, \bar{\theta}_j)\bar{\theta} &= \bar{D}(q)\epsilon(t_k) \\ \frac{\partial \hat{\varphi}_f(t_k, \hat{\theta}_j)}{\partial \hat{\theta}_j} &= fp(u(t_k)) \end{aligned}$$

where fp is some function of $u(t)$ and from assumption-2, it can be observed that these two terms are asymptotically independent. In the same way it can also be proved that the second term is also independent (see equations after 24 in [10]). Thus ,in the asymptotic case, $\hat{\theta}_{j+1}$ is locally convergent to $\bar{\theta}$. This completes the theorem

5.2 Proofs of Greedy Sparse Algorithm

As explained in chapter-4, it has to be proven that any optimal point is a CW-minima point [9].
Instead of proving it, one can also prove that any CW-minima point is a BF-vector and any optimal point is a BF-vector.

Theorem 5.2.1. *Let $x^* \in C_s$ be a optimal point of $\min\limits_{||\theta||_0 < s} f(\theta)$, then x^* is a BF-vector.*

Proof. If $||x^*|| < s$ and let $g(t) = f(x^* + te_i)$
As x^* is an optimal point $\Rightarrow\ 0\ \in\ argmin\{g(t)\}$
otherwise, $\exists$ a t_0 for which, $f(x^* + t_0 e_i) < f(x^*)$
when $t = 0$, there will be a minimum and thus $g'(t) = 0 \Rightarrow g'(t) = 0$

$\Rightarrow \bigtriangledown_i f(x^*) = 0 \;\forall\; i \in \{1, 2, \ldots, n\} \;\Rightarrow\; \bigtriangledown f(x^*) = 0$

If $||x^*|| = s$, the same argument holds good but in this case, our $i \in I_1(x^*) \Rightarrow \bigtriangledown_i f(x^*) = 0$ □

Theorem 5.2.2. *Let $x^* \in C_s$ be a CW-minima of $\min\limits_{||\theta||_0 < s} f(\theta)$, then x^* is a BF-vector.*

Proof. When $||x^*|| = s, i \in I_1(x^*)$ and $j = 1, 2, \ldots, n$, and x^* is CW-Minima,

$\Rightarrow$ by definition, $f(x^*) \leq \min\limits_{t\in\mathbb{R}} f(x^* - x_i^* e_i + te_j)$

If $i = j$, $\Rightarrow f(x^*) \leq \min\limits_{t\in\mathbb{R}} f(x^* - x_i^* e_i + te_i)$, as x^* is CW-minima point

$\Rightarrow$ $f(x^*)$ is the minimum point, so $f(x^*) = \min\limits_{t\in\mathbb{R}} f(x^* - x_i^* e_i + te_i)$

Let $s = t - x_i^* \;\Rightarrow\; f(x^*) = \min\limits_{s\in\mathbb{R}} f(x^* + se_i)$ and if x^* is CW-Minima point then by the same argument of the above theorem, it can be observed that x^* is BF-vector. □

References

[1] H. Garnier, M. Mensler, and A. Richard. Continuous-time model identification from sampled data: implementation issues and performance evaluation. *International Journal of Control* 76, (2003) 1337–1357.

[2] H. Unbehauen and G. Rao. A review of identification in continuous-time systems. *Annual reviews in Control* 22, (1998) 145–171.

[3] P. Jampana and K. Detroja. Continuous Time Identification in Laplace Domain. In Dynamics and Control of Process Systems, volume 10. 2013 367–372.

[4] N. K. Sinha. Identification of continuous-time systems from samples of input-output data: An introduction. *Sadhana* 25, (2000) 75–83.

[5] J. O. Ramsay, G. Hooker, D. Campbell, and J. Cao. Parameter estimation for differential equations: a generalized smoothing approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69, (2007) 741–796.

[6] H. Garnier, L. Wang, and P. C. Young. Direct identification of continuous-time models from sampled data: Issues, basic solutions and relevance. Springer, 2008.

[7] P. C. Young, H. Garnier, and M. Gilson. Refined instrumental variable identification of continuous-time hybrid Box-Jenkins models. In Identification of continuous-time models from sampled data, 91–131. Springer, 2008.

[8] A. Poyton, M. S. Varziri, K. B. McAuley, P. McLellan, and J. O. Ramsay. Parameter estimation in continuous-time dynamic models using principal differential analysis. *Computers & chemical engineering* 30, (2006) 698–708.

[9] A. Beck and Y. C. Eldar. Sparsity constrained nonlinear optimization: Optimality conditions and algorithms. *SIAM Journal on Optimization* 23, (2013) 1480–1509.

[10] X. Liu, J. Wang, and W. X. Zheng. Convergence analysis of refined instrumental variable method for continuous-time system identification. *Control Theory & Applications, IET* 5, (2011) 868–877.

[11] P. Young. Parameter estimation for continuous-time models—a survey. *Automatica* 17, (1981) 23–39.

[12] G. Rao and H. Unbehauen. Identification of continuous-time systems. In Control Theory and Applications, IEE Proceedings-, volume 153. IET, 2006 185–220.

[13] G. Hooker. Theorems and calculations for smoothing-based profiled estimation of differential equations. Technical Report, Technical Report BU-1671-M, Department of Biological Statistics and Computational Biology, Cornell University, Ithaca, NY 2007.

[14] T. D. Söderström and P. G. Stoica. Instrumental variable methods for system identification, volume 57. Springer, 1983.

[15] T. Söderström and P. Stoica. System identification. Prentice-Hall, Inc., 1988.