University of Alberta

State and Parameter Estimation for LPV Systems

by

Ying Wang

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Control Systems

Department of Electrical and Computer Engineering

Fall 2012
Edmonton, Alberta

To my parents and family

Abstract

This thesis develops an online parameter and state estimation scheme for a linear parameter varying (LPV) system that utilizes an iterative moving window technique. In the proposed scheme, an online algorithm, based on the input/output measurement, is implemented to approximate the real system with the best match LPV model. The varying parameters in the LPV model can be estimated by solving a quadratic programming optimization problem, and state variable values can be calculated with an adaptive state observer. As an application, the wind turbine system is formulated as an LPV model and applied by the proposed scheme. In addition, the ranges of state and uncertainty are obtained in an online fault detection (FD) scheme, based on parity space models using a technique similar to the iterative moving estimation window. A two-level adaptive threshold for FD is designed to decrease the miss alarm rate based on the estimated ranges.

Acknowledgements

I would like to extend my sincere thanks to my supervisor, Dr. Qing Zhao, for her pivotal role, guidance, constant support, patience and encouragement throughout my whole research work.

I also thank the members of my thesis committee, Dr. Tongwen Chen and Dr. Vinay Prasad, for their valuable comments and advices in improving this thesis.

Many thanks to my friends, Russell Dodd, Ruoshui Ni, Hui Wang for their time and effort on my thesis writing and my colleagues in the control system group, such as Shuonan Yang, Xiangyu Meng, Geoff MacDonald, Dawei Shi for their help and support in various aspects of my research. I feel deeply honored and pleased to work with these intelligent colleagues in this great research team.

My sincere appreciation and gratitude are dedicated to my family for their constant love, support and encouragement throughout my stay in Edmonton.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

It is desirable to know the state variables in the design and operation of control systems; however, measuring them is not always practical. A state observer provides an estimate of the internal state of a system. For example, the Bayesian observer can deal with the state estimation in a linear time-invariant (LTI) model with stochastic noises. The classical Kalman filter is usually applied in the LTI state space model with Gaussian process noise and Gaussian measurement noise [1] [2] [3]. However, it is difficult to use standard optimization techniques for LTI systems in linear parameter varying (LPV) systems because the parameters are unknown and constantly changing. Nevertheless, in industry, under limited conditions with known input/output signals and unknown model parameters, the state estimation of an LPV or nonlinear system is still required for the purposes of control and supervision. In this thesis, an online scheme for an LPV system is designed to estimate the state variables and system parameters in two different sliding estimation windows. The state estimation is based on parameter values, estimated in the parameter estimation window, while the parameter estimation utilizes the state variable values that are estimated in the previous state estimation window.

A wind turbine system is chosen as a pilot test bench to verify the effectiveness of the state estimation scheme for LPV systems. According to a typical control strategy, a wind turbine system usually has three operating modes. Thus, it can be described as different models with different model parameters based on the three different modes. Through the proposed scheme, states and parameters in the wind turbine model can be obtained. This thesis describes an online state and parameter estimation scheme described to estimate unknown parameters using measured input signal (wind velocity) and output (generated power) signals. A complete simulation is presented in Chapter 3 which also includes system modeling and result analysis. This simulation shows the validity of the scheme based on a Simulink wind turbine benchmark.

An online fault detection scheme for an LTI system with polytopic uncertainty is designed using a technique similar to the estimation sliding windows. It is well known that model-based fault detection generates a residual that can be checked using a threshold to distinguish between faulty and nominal cases. In simple cases, the threshold can be selected by the user as certain constant values, while in systems with model uncertainties or disturbances, threshold generation can be treated as a separate design process from residual generation, and the threshold will depend on the ranges of states and uncertainties, which are not always available but can be realized by quantitative analysis of the uncertainties. As the states and uncertainties vary in each detection window, an adaptive threshold is designed for the online fault detection.

1.2 Outline and Contribution of the Thesis

The remainder of this chapter provides a literature review which includes the related concepts of moving horizon state estimation, state estimation for LPV systems, and model predictive control. The above concepts are introduced to develop the algorithms for the state and parameter estimation scheme and the online fault detection algorithm.

Chapter 2 demonstrates a state and parameter iterative estimation scheme for an LPV system. The contribution of this chapter lies in developing an online scheme integrating the optimization and state observer methods, and the related convergence analysis.

In Chapter 3, the algorithm in Chapter 2 is implemented in wind turbine models. A typical wind turbine system is introduced, and a benchmark test model is described in detail. The key problem involves the selection of operating points and linearization of a nonlinear function in the development of local linear models. However, the linearized local models of the wind turbine system are continuous ones, and system matrices are numerically ill-conditioned, so the zero-order hold discretization and similar transformation techniques are applied. Nevertheless, the discretization and transformation result in a biased calculation of the estimated parameters. Therefore, a procedure that recovers the parameters in real models from the parameters in the scheme is discussed. Finally, the simulation results are obtained and analyzed. A complete application of a real wind turbine system using the state and parameter estimation scheme is provided in this chapter.

In Chapter 4, a robust fault detection algorithm involving state and uncertainty estimation based on a parity space model is proposed. This chapter focuses on threshold generation and selection, which can indicate the nominal and faulty situations. The false alarm rate is decreased using a two-level threshold rather than a single constant value as the threshold.

Chapter 5 provides a conclusion of the thesis and suggestions for future research.

1.3 Literature Review

This section provides a short review of the related research on moving horizon estimation (MHE), state estimation for LPV systems, and model predictive control.

1.3.1 Moving Horizon State Estimation

In a control system, the real state in the system cannot usually be measured directly; however, state estimation can be used to estimate the state values and reduce the effect of measurement noise and disturbance.

Based on a state space model with deterministic or stochastic noise, the state estimation methods differ. If the disturbance or noise can be neglected, a convergent iterative algorithm named a state observer is established based on the measurement of inputs and outputs, such as a classical Luenberger observer [4] or an extended Luenberger observer [5]. If the probabilistic information of the noise is known, the state estimation can be considered an optimization problem involving the probability density function. The most well-known observers are the Kalman filter for the linear model [6] and the extended Kalman filter [7] for the nonlinear model.

State estimation methods can be grouped as batch state estimation and moving horizon estimation types. Batch state estimation optimizes the cost function off-line, which includes all the history and current measured information. Moving horizon state estimation provides an online solution by minimizing a cost function over a previous horizon window. As the horizon window is sliding, the updated measurement signals are supplied in a new optimization problem, which can be used to improve the result of the state estimation. In order to avoid data redundancy, a fixed-length horizon window is widely used. Sometimes the optimization also includes some constraints; hence, the optimization technique with constraints is also applied in each sliding window.

Moving horizon estimation can be applied in cases with either of the probabilistic and deterministic models. From an engineering point of view, since it allows the easy addition of constraints to the optimization problem without changing the structure, MHE can be widely used in industry. A Kalman filter provides a recursive solution, which can be considered as a moving horizon state estimation typically used in online control.

1.3.2 State Estimation for LPV Systems

The real physical systems are usually nonlinear; nevertheless, LPV models can be considered to approximate the nonlinear systems. Therefore, more and more researchers have paid attention to state estimation for LPV systems, which is vital for both fault diagnosis and control design.

A design of the fault diagnosis for an LPV system is represented in [8–14]. In these studies, the LPV system description is used to approximate the behavior of a nonlinear system, and a residual is generated for an LPV polytopic system. This residual is robust against a disturbance and sensitive to the faults.

In both the control design and fault diagnosis, the LPV systems are usually obtained by using Jacobian linearization, which yields several linearized models around the operating points [8–11] [13–15]. The linearized models are the basis of the LPV systems and are called linear local models in this scheme. When the parameters in the LPV systems do not depend on the external variables and do not need Jacobian linearization, this form is named a quasi-LPV system [10,12,16,17]. Furthermore, the state estimation methods for control design and fault diagnosis in the linear systems are used to generate a family of linear state estimators corresponding to the linear local models of an LPV system. In the polytopic LPV system, a class of varying parameters is designed and assigned to the family of the linear local models for the best control results. For instance, the assigned parameters for fault diagnosis should be sensitive to the fault, and robust against a disturbance and an uncertainty; for control design, the closed loop system stability or the desired performance with estimated state feedback should be guaranteed.

1.3.3 Model Predictive Control

Model predictive control (MPC) is a reliable technique and is used diffusely in the chemical, food processing, automotive, and aerospace fields. It is based on the linear discrete time state space model and formulated by introducing an open-loop optimization, in which the objective function is often achieved with a quadratic programming over a control horizon and a prediction horizon for the purpose of the algorithm and commercial software [18]. If both of the control and prediction horizons approach infinity and the optimization contains no constraint, the MPC problem can be converted into a standard linear quadratic regulator (LQR) problem

[19]. However, when the two horizons are finite, MPC becomes a step-by-step optimization method, which also can handle the constraints on the inputs and outputs in each step. At each estimation window, a new control signal value is calculated to update the optimization problem as a new initial condition on the basis of the current measurement and the prediction of the states [18]. This process shifts and repeats.

MPC predicts the future behavior using control algorithms. It attempts to optimize future behavior by computing a sequence of future manipulated variable adjustments in a horizon window. MPC is based on an iterative finite horizon optimization, so that the prediction horizon keeps being shifted forward. As soon as the prediction horizon window moves, a new optimization is processed based on the updated measurement. However, MPC can deal with system noise but has difficulty in handling the explicit model uncertainties [20, 21]. Therefore, more and more researchers paid attention to the robust MPC [22–26].

The difference between MPC and MHE is that MHE uses an open loop scheme to estimate the current state variable, while MPC utilizes a closed loop control algorithm to predict the future behavior. They both are based on an online state space model optimization over a sliding horizon window.

Chapter 2

State and Parameter Estimation for an LPV System

2.1 Introduction

The LPV model has received attention from researchers in the last few decades since it can bridge Linear Time Invariant (LTI) models and nonlinear models. The methods of converting from nonlinear to LPV models are usually open loop algorithms under a static state feedback, and this algorithm does not have any effects on the real system [27, 28]. The nonlinear model can be linearized as a collection of linear models around several operating points. The generation of an LPV models starts from the linearized state space models, which describe the local behaviors, and then is combined with a set of varying variable vectors to represent the nonlinear system.

In this chapter, an online scheme is designed to estimate the unknown varying parameters and state variables within different sliding windows. The parameter estimation algorithm for LPV systems is based on estimated state variables from the previous window, while the state estimation is based on estimated parameters of the LPV model. The two estimations are carried out iteratively so as to form a practical online estimation scheme. In the parameter estimation algorithm, the problem is established as a quadratic programming optimization problem which can be solved by the least square technique, and the state estimation algorithm utilizes an adaptive Luenberger observer. In a relatively large state estimation window, the model with estimated parameters can be considered known, and state estimation errors are nonincreasing thanks to the adaptive Luenberger observer. Parameters in the LPV system are calculated by solving a quadratic optimization problem, and they are more accurate because the parameter estimation depends on more precise estimated state values.

2.2 Problem Formulation

For a nonlinear system $\dot{x} = f(x,\, u),\ y = g(x,\, u)$, the p operating points $\{x_{o1}, x_{o2}, \cdots, x_{op}\}$ at the p input $\{u_{o1}, u_{o2}, \cdots, u_{op}\}$ satisfy the state equation as follows according to the definition of operating points [29],

$$\dot{x} = f(x,\, u)|_{x=x_{oi}, u=u_{oi}} = f(x_{oi},\, u_{oi}) = 0,$$

where u_{oi} is called a local input and x_{oi} is an operating point.

According to the linearization of multi variables principle

$$f(x,y) = f(a,b) + \frac{\partial f(x,y)}{\partial x}|_{\substack{x=a\\y=b}}(x-a) + \frac{\partial f(x,y)}{\partial y}|_{\substack{x=a\\y=b}}(y-b),$$

the nonlinear system can be linearized to

$$\begin{array}{l}\dot{x} = f(x,u) = f(x_{oi}, u_{oi}) + \frac{\partial f(x,u)}{\partial x}|_{\substack{x=x_{oi}\\u=u_{oi}}}(x - x_{oi}) + \frac{\partial f(x,u)}{\partial u}|_{\substack{x=x_{oi}\\u=u_{oi}}}(u - u_{oi}),\\ y = g(x,u) = g(x_{oi}, u_{oi}) + \frac{\partial g(x,u)}{\partial x}|_{\substack{x=x_{oi}\\u=u_{oi}}}(x - x_{oi}) + \frac{\partial g(x,u)}{\partial u}|_{\substack{x=x_{oi}\\u=u_{oi}}}(u - u_{oi}),\end{array}$$

where $f(x_{oi}, u_{oi}) = 0$. These linearized models are called linear local models. After adding all the constants up to ϕ_1 and ϕ_2 in each equation, one of the obtained linear local models can be reformed as

$$\begin{array}{l}\dot{x} = A_{ci}x + B_{ci}u + \phi_1\\ y = C_{ci}x + D_{ci}u + \phi_2,\end{array} \tag{2.1}$$

where $A_{ci} = \frac{\partial f(x,u)}{\partial x}|_{x=x_{oi}u=u_{oi}}$, $B_{ci} = \frac{\partial f(x,u)}{\partial u}|_{x=x_{oi},u=u_{oi}}$, $C_{ci} = \frac{\partial g(x,u)}{\partial x}|_{x=x_{oi},u=u_{oi}}$, $D_{ci} = \frac{\partial g(x,u)}{\partial u}|_{x=x_{oi},u=u_{oi}}$, $\phi_1 = \frac{\partial f(x,u)}{\partial x}|_{x=x_{oi},u=u_{oi}}x_{oi} + \frac{\partial f(x,u)}{\partial u}|_{lx=x_{oi},u=u_{oi}}u_{oi}$, $\phi_2 = g(x_{oi}, u_{oi}) + \frac{\partial g(x,u)}{\partial x}|_{x=x_{oi}u=u_{oi}}x_{oi} + \frac{\partial g(x,u)}{\partial u}|_{x=x_{oi}u=u_{oi}}u_{oi}$.

The model can be reformulated as a standard state space model after constructing the terms of input and constant to a matrix $\begin{bmatrix} u & 1 \end{bmatrix}^T$,

$$\begin{aligned} \dot{x} &= A_{ci}x + \begin{bmatrix} B_{ci} & \phi_1 \end{bmatrix} \begin{bmatrix} u \\ 1 \end{bmatrix} = A_{ci}x + B_{cmi}U \\ y &= C_{ci}x + \begin{bmatrix} D_{ci} & \phi_2 \end{bmatrix} \begin{bmatrix} u \\ 1 \end{bmatrix} = C_{ci}x + D_{cmi}U. \end{aligned} \tag{2.2}$$

In the linear local model, the system matrices depend on the selection of operating points. The constructed model (2.2) maintains the controllability and observability of the original model (2.1).

Using zero-order hold discretization with a sampling time of T_s, the continuous time model (2.2) can be converted to a discrete time state space model

$$\begin{aligned} &x(n+1) = A_i x(n) + B_i U(n) \\ &y(n) = C_i x(n) + D_i U(n), \end{aligned} \tag{2.3}$$

where n is the current time index, $A_i = e^{A_{ci}T}$, $B_i = \int_{\tau=0}^{T} e^{A_{ci}\tau} B_{cmi} d\tau = A_{ci}^{-1}(A_i - I)B_{cmi}$, $C_i = C_{ci}$, $D_i = D_{cmi}$.

The discrete time linear local model (2.3) is adopted in the proposed algorithm. The selection of linear local models highly depends on choice of operating points, which are usually based on the knowledge of input signals.

A nonlinear model can be converted to an LPV model with unknown varying parameters $P^f(n)$ on the basis of linear local models, as shown in Equation (2.4). The system matrices are of polytopic type that represents a linear combination of the linear local models with respect to these unknown weighting parameters.

$$\begin{aligned} &x(n+1) = A(P^f(n))x(n) + B(P^f(n))U(n) \\ &y(n) = C(P^f(n)) \cdot x(n) + D(P^f(n)) \cdot U(n) \end{aligned} \tag{2.4}$$

where $A(P^f(n)) = \sum_{i=1}^{p} P_i^f(n) \cdot A_i$, $B(P^f(n)) = \sum_{i=1}^{p} P_i^f(n) \cdot B_i$, $C(P^f(n)) = \sum_{i=1}^{p} P_i^f(n) \cdot C_i$, $D(P^f(n)) = \sum_{i=1}^{p} P_i^f(n) \cdot D_i$, $\sum_{i=1}^{p} P_i^f(n) = 1$ and $0 \leq P_i^f(n) \leq 1$, $y(n)$ is the real output signal, $U(n)$ is the constructed input signal that $U(n) = \begin{bmatrix} u(n) & 1 \end{bmatrix}^T$.

There are three assumptions listed as follows:

Assumption 2.1

The time varying parameter $P^f(n) = \begin{bmatrix} P_1^f(n) & \cdots & P_p^f(n) \end{bmatrix}^T$ keeps constant as $P(k)$ when $kM \leq n \leq (k+1)M$.

The LPV model (2.4) is rewritten as follows, when $kM \leq n \leq (k+1)M$,

$$\begin{array}{l} x(n+1) = A(P(k))x(n) + B(P(k))U_n \\ y(n) = C(P(k))x(n) + D(P(k))U(n) \end{array} \tag{2.5}$$

where $A(P(k)) = \sum_{i=1}^{p} P_i(k) \cdot A_i$, $B(P(k)) = \sum_{i=1}^{p} P_i(k) \cdot B_i$, $C(P(k)) = \sum_{i=1}^{p} P_i(k) \cdot C_i$, $D(P(k)) = \sum_{i=1}^{p} P_i(k) \cdot D_i$, $\sum_{i=1}^{p} P_i(k) = 1$ and $0 \leq P_i(k) \leq 1$. In process control it exists diffusely the slow rate LPV systems with slow varying parameters.

Assumption 2.2

All the linear local models are stable.

It means that $x_{next}(i,n) = A_i x(n) + B_i U(n), y_{next}(i,n) = C_i x(n) + D_i U(n)$ are bounded in a estimation window $kM \leq n \leq (k+1)M$.

Assumption 2.3

The system in (2.5) is observable.

2.3 Algorithm for State and Parameter Estimation

The algorithm for state and parameter estimation uses two sliding estimation windows iteratively, one of which is for state estimation under the condition of the estimated parameters, while the other is for parameter estimation using an optimization technique based on the estimated state values from the previous step. This open loop algorithm allows an easy implementation without any interruption of the original system.

According to the aforementioned techniques, the procedure of the proposed parameter and state estimation scheme can be divided into three steps: initialization, state estimation, and parameter estimation.

2.3.1 Initialization

At the beginning of the algorithm, the initial parameters can be calculated using the initial input and output signals through the steady-state curve method [29].

The system steady state curve is shown as a one-to-one mapping in an input-output graph. Figure 2.1 illustrates the steady state curves of several single input single output (SISO) systems. A nonlinear model is represented as a curve, while the linearized model around a particular operating point is plotted as a tangent line across it.

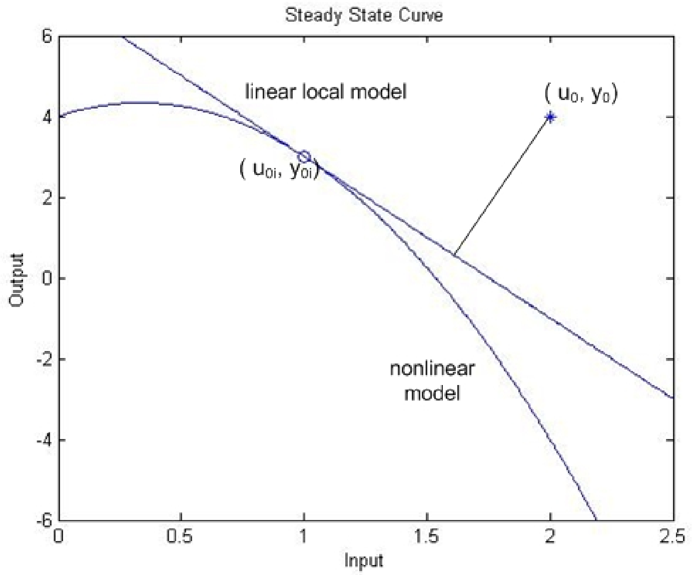


Figure 2.1: Relation between nonlinear systems and linearized systems

Any point (u, y) on a tangent line that crosses the point (u_{oi}, y_{oi}) satisfies

$$y = y_{oi} + C_i(I - A_i)^{-1}B_i(u - u_{oi}),$$

where (u_{oi}, y_{oi}) is the operating point of the linearized model and called respectively as local input and local output.

The tangent lines can be expressed as $y = G_i u + M_i$, where $G_i = C_i(I - A_i)^{-1}B_i$, $M_i = y_{oi} - C_i(I - A_i)^{-1}B_i u_{oi}$. The measured input and output at the initial time are denoted as (u_0, y_0). The shortest distance from the measured input/output signal point to a linear local model line in Figure 2.1 is expressed as

$$\min \|y - y_0\|_2^2 + \|u - u_0\|_2^2.$$

According to the convex optimization theory, the point in the tangent line closest to

(u_0, y_0) is

$$u_{v,i} = (G_i^T G_i + I)^{-1}(G_i^T y_0 + u_0 - G_i^T M_i)$$

$$y_{v,i} = G_i u_{v,i} + M_i.$$

where I is the identity matrix with proper dimension. The shortest distance from (u_0, y_0) to all the tangent lines is given as

$$\min_i \|y_{v,i} - y(0)\|_2^2 + \|u_{v,i} - u(0)\|_2^2.$$

When the optimal i^{th} linear local model is chosen, $\hat{P}_i(-1)$ is defined as 1 and the other elements of vector $\hat{P}(-1)$ are 0. In the initialization step the i^{th} linear local model is pre-selected based on the measured initial input/output signals so that the initial parameter $\hat{P}(-1)$ has been estimated.

2.3.2 State Observer

In the proposed scheme, the state observer method is used for estimating the states, while an optimization technique is employed for the parameter estimation. They are implemented in an online iterative algorithm based on discrete time models. There are three indexes in this algorithm such as the horizon window index k, the discrete time model index n and the continuous time index t.

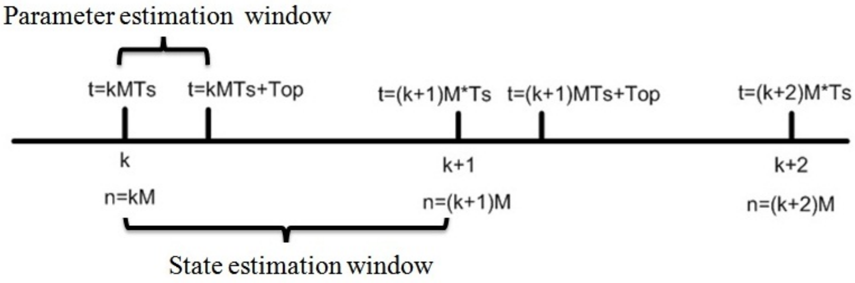


Figure 2.2: Relation between state estimation window and parameter estimation window

It is assumed that the parameter estimation time is much shorter than the state estimation time. As shown in Figure 2.2, in the k^{th} horizon window, the parameter estimation window starts from $t = kMT_s$ and ends at $t = kMT_s + T_{op}$, while the state estimation window begins from $t = kMT_s$ and ends at $t = (k+1)MT_s$, where

T_s is the sampling time and T_{op} is the optimization time for parameter estimation. The assumption indicates $T_{op} << MT_s$, which results from the modern computer processors' power in solving optimization problems. $T_{op} << MT_s$ also can be restricted to $T_{op} < T_s$ to guarantee the parameter and state estimation windows from overlapping.

A state observer can provide state estimation when the system model and measurement of input/output signal are given. In the state estimation window with horizon index k, the unknown parameters $P(k)$ in the system model can be estimated as $\hat{P}(k)$ from the parameter estimation window with horizon index k. When the system model with estimated parameter $\hat{P}(k)$ and measured input/output signal are obtained, the estimated states using a designed state observer are approaching the real state values within M steps in the state estimation window.

In this scheme, the classical Luenberger observer is employed due to its simple structure and the fact that the noise is not considered in the system. However, a different state estimator can be chosen when considering the improved performance, such as rejection of measurement noises, or robustness towards parameter estimation errors, etc. Kalman filters and/or the robust state observers are possible choices in this case. In the state estimation window with horizon index k, the state observer is designed as

$$\begin{cases} \hat{x}(n+1) = A(\hat{P}(k))\hat{x}(n) + B(\hat{P}(k))U(n) + L(k)(y(n) - \hat{y}(n)) \\ \hat{y}(n) = C(\hat{P}(k))\hat{x}(n) + D(\hat{P}(k))U(n) \end{cases} \tag{2.6}$$

where $A(\hat{P}(k))=\sum_{i=1}^{p} \hat{P}_i(k)A_i$, $B(\hat{P}(k))=\sum_{i=1}^{p} \hat{P}_i(k)B_i$, $C(\hat{P}(k))=\sum_{i=1}^{p} \hat{P}_i(k)C_i$, $D(\hat{P}(k))=\sum_{i=1}^{p} \hat{P}_i(k)D_i$, $\sum_{i=1}^{p} \hat{P}_i(k)=1$ and $0 \leq \hat{P}_i(k) \leq 1$. The initial parameter $\hat{P}(-1)$ is decided in the initialization step, and the initial state variable $\hat{x}(0)$ is the operating point x_{oi} in the optimal i^{th} linear local model from the initialization step. $\hat{P}(k)$ is obtained from the parameter estimation window. $L(k)$ is designed so that the eigenvalues of the closed loop system matrix $A(\hat{P}(k))) + L(k)\hat{C}(k)$ are inside of the unit circle in z-plane. When these eigenvalues are closer to the origin in z-plane, the estimated states will approach the real values faster.

2.3.3 Parameter Estimation

The parameter estimation with horizon index $(k{+}1)$ is carried out based on the states $\hat{x}(kM{+}M)$, that is estimated from the state estimation window with horizon index k. As shown in Figure 2.2, the parameter estimation window starts from $t{=}(k{+}1)MT_s$ and ends at $t{=}(k{+}1)MT_s{+}T_{op}$. The parameter estimation is realized by solving an optimization problem as follows

$$\min_{\hat{P}(k+1)} \psi^T(kM+M)\Sigma_1\psi(kM+M) + \Psi^T(kM+M)\Sigma_2\Psi(kM+M) \qquad (2.7)$$

subject to

$$\begin{aligned} &y(k{+}M){=}C(\hat{P}(k{+}1))\hat{x}(kM{+}M){+}D(\hat{P}(k{+}1))U(kM{+}M){+}\psi(kM{+}M)\\ &\tilde{x}(kM{+}M{+}1){=}A(\hat{P}(k{+}1))\hat{x}(kM{+}M){+}B(\hat{P}(k{+}1))U(kM{+}M){+}\Psi(kM{+}M), \end{aligned}$$

where the predicted state variable is expressed as

$$\begin{aligned} \tilde{x}(kM{+}M{+}1) \;&= A(\hat{P}(k))\hat{x}(kM{+}M) + B(\hat{P}(k))U(kM{+}M) + L(k)\\ &[y(kM{+}M) - C(\hat{P}(k))\hat{x}(kM{+}M) - D(\hat{P}(k))U(kM{+}M)]. \end{aligned}$$

In the expression of the predicted state variable $\tilde{x}(kM{+}M{+}1)$, $\hat{P}(k)$ and $L(k)$ are calculated from the parameter estimation window with horizon window k. In the constraint equations, the system matrices are expressed in the polytopic forms with respect to the desirable parameter vector $\hat{P}(k{+}1)$: $A(\hat{P}(k{+}1)){=}\sum_{i=1}^{p}\hat{P}_i(k{+}1)A_i$, $B(\hat{P}(k{+}1)){=}\sum_{i=1}^{p}\hat{P}_i(k{+}1)B_i$, $C(\hat{P}(k{+}1)){=}\sum_{i=1}^{p}\hat{P}_i(k{+}1)C_i$ and $D(\hat{P}(k{+}1)){=}\sum_{i=1}^{p}\hat{P}_i(k{+}1)D_i$. The input and output signals $U(kM{+}M)$ and $y(kM{+}M)$ are measured from the real system. In the objective function, Σ_1 and Σ_2 are the weighting matrices.

In the optimization problem, the product terms with above polytopic forms in the constraint equations, $A(\hat{P}(k{+}1))\hat{x}(kM{+}M)$, $B(\hat{P}(k{+}1))U(kM{+}M)$, $C(\hat{P}(k{+}1))\hat{x}(kM{+}M)$, $D(\hat{P}(k{+}1))U(kM{+}M)$, can be represented as the matrix products of the optimized variable $\hat{P}(k{+}1)$, one of which is represented as

$$\begin{aligned} &C(\hat{P}(k{+}1))\hat{x}(kM{+}M) = \sum_{i=1}^{p}\hat{P}_i(k{+}1)C_i\hat{x}(kM{+}M)\\ &= \begin{bmatrix} C_1\hat{x}(kM{+}M) & \cdots & C_p\hat{x}(kM{+}M) \end{bmatrix} \begin{bmatrix} \hat{P}_1(k{+}1)\\ \vdots \\ \hat{P}_p(k{+}1) \end{bmatrix} = \hat{C}_{px}(kM{+}M)\hat{P}(k{+}1). \end{aligned}$$

Similarly the terms $A(\hat{P}(k{+}1))\hat{x}(kM{+}M)$, $B(\hat{P}(k{+}1))U(kM{+}M)$ and $D(\hat{P}(k{+}1))U(kM{+}M)$ can be expressed respectively as $\hat{A}_{px}(kM{+}M)\hat{P}(k{+}1)$, $B_{pu}(kM{+}M)\hat{P}(k{+}1)$ and $D_{pu}(kM{+}M)\hat{P}(k{+}1)$, where

$$\begin{aligned}\hat{A}_{px}(kM{+}M) &= \begin{bmatrix} A_1\hat{x}(kM{+}M) & \cdots & A_p\hat{x}(kM{+}M)\end{bmatrix},\\ B_{pu}(kM{+}M) &= \begin{bmatrix} B_1U(kM{+}M) & \cdots & B_pU(kM{+}M)\end{bmatrix},\\ D_{pu}(kM{+}M) &= \begin{bmatrix} D_1U(kM{+}M) & \cdots & D_pU(kM{+}M)\end{bmatrix}.\end{aligned}$$

The constraint equations are transformed to the linear ones with respect to optimized vector $\hat{P}(k{+}1)$, thus the optimization problem becomes a standard quadratic programming optimization problem (2.7) that can be solved by the least square technique. When the parameter $\hat{P}(k+1)$ is acquired in the parameter estimation with horizon index $(k+1)$, it will be considered as a known condition in the state estimation window with horizon index $(k+1)$. The parameter estimation alternates with the state estimation, and so forth, which forms an online parameter and state estimation scheme.

2.4 Convergence Analysis

Convergence analysis is carried out using the Lyapunov method for stability analysis. In the estimation theory for discrete-time systems, if there exists a Lyapunov function such that $V(e(k)) > 0$ and $V(e(k+1)) - V(e(k)) < 0$, then $\lim_{k\to\infty} e(k) = 0$, where $e(k)$ is the estimation error.

On the basis of the two sliding windows' illustration, the parameter estimation is realized by one-step optimization in the time of T_{op}, so that the parameter estimation error is minimum in one horizon window. Therefore, the convergence analysis on state estimation error needs to be performed in a horizon window with index k.

There exists a lemma based on Assumption 2.2 before the analysis on state estimation error.

Lemma 2.1 The defined polynomial $\eta(k,n) = A_{px}(n) + B_{pu}(n) - L(k)C_{px}(n) - L(k)D_{pu}(n)$ is norm bounded, $||\eta(k,n)||_2 \le \gamma(k)$, in a horizon window $kM \le n \le (k+1)M$, where $\gamma(k)$ is a positive scalar.

PROOF

In light of the definition that

$$\begin{aligned} A_{px}(n) &= \begin{bmatrix} A_1x(n) & \cdots & A_px(n) \end{bmatrix}, \quad & C_{px}(n) &= \begin{bmatrix} C_1x(n) & \cdots & C_px(n) \end{bmatrix}, \\ B_{pu}(n) &= \begin{bmatrix} B_1U(n) & \cdots & B_pU(n) \end{bmatrix}, \quad & D_{pu}(n) &= \begin{bmatrix} D_1U(n) & \cdots & D_pU(n) \end{bmatrix} \end{aligned}$$

and $\eta(k,n)$ can be written as a matrix made up with p column vectors, such as $\eta(k,n) = \begin{bmatrix} \eta_1(k,n) & \cdots & \eta_p(k,n) \end{bmatrix}$, where $\eta_i(k,n){=}A_ix(n){+}B_iU(n){-}L(k)C_ix(n){-}L(k)D_iU(n) = x_{next}(i,n){-}L(k)y_{next}(i,n)$.

The condition, $x_{next}(i,n)$ and $y_{next}(i,n)$ are bounded, is implied in Assumption 2.2, and $L(k)$ is a constant matrix within one horizon window. So the column vector $\eta_i(k,n) = x_{next}(i,n) - L(k)y_{next}(i,n)$ is bounded, which implies $\eta(k,n)$ is norm bounded $||\eta(k,n)||_2 \leq \gamma(k)$. Therefore lemma is proved.

Assume that the parameter estimation error $e_p(k) = P(k) - \hat{P}(k)$ satisfies a condition that

$$||e_p(k)||_2^2 \leq \alpha(k)||\epsilon(n)||_2^2$$

in the estimation window $kM{\leq}n{\leq}(k{+}1)M$, where $\alpha(k)$ is a positive scalar.

Theorem 2.1 If there exists the positive scalars α, ω, γ, positive semi-definite matrices Q and S such that the following conditions are satisfied,

$$F^TQF - Q + \tfrac{1}{\omega}F^TF + \alpha\gamma(||Q^{0.5}||_2^2 + \omega||Q||_2^2) + S = 0 \qquad (2.8)$$

in the horizon window of $kM \leq n \leq (k+1)M$, then the state estimation error $\epsilon(n)$ are nonincreasing.

In the equation (2.9), $F{=}A(\hat{P}(k)){-}L(k)A(\hat{P}(k))$, e_p is the parameter estimation error, $||\bar{\epsilon}||_2$ is the upper norm bound on state estimation error, and $\bar{\eta}$ is the upper bound on $\eta(k,n)$ in the aforementioned lemma.

PROOF

An existence of a quadratic Lypapunov function $V(n)$ which satisfies that

$$V(n+1) - V(n) = \epsilon^T(n+1)Q\epsilon(n+1) - \epsilon^T(n)Q\epsilon(n) \leq 0$$

assures the state estimation errors nonincreasing, where Q is the positive symmetric matrix.

The real state variables $x(n)$ in one horizon window $kM \leq n \leq (k+1)M$ satisfy the LPV model (2.5), while the estimated state variables are calculated in the Luenberger observer (2.6). Then the state variable error $\epsilon(n+1) = x(n+1) - \hat{x}(n+1)$ can be expressed as

$$\epsilon(n{+}1){=}(A(\hat{P}(k)){-}L(k)C(\hat{P}(k)))\epsilon(k){+}(A(P(k)){-}A(\hat{P}(k)))x(n){+}(B(P(k)){-}B(\hat{P}(k)))U(n){-}L(k)(C(P(k)){-}C(\hat{P}(k)))x(n){-}L(k)(D(P(k)){-}D(\hat{P}(k)))U(n).$$

Following the same linear transformation in the parameter estimation step, some terms in the above expression can be converted to $A(P(k))x(n) = A_{px}(n)P(k)$, $B(P(k))U(n) = B_{pu}(n)P(k)$, $C(P(k))x(n) = C_{px}(n)P(k)$, $D(P(k))U(n) = D_{pu}(n)P(k)$, $A(\hat{P}(k))x(n) = A_{px}(n)\hat{P}(k)$, $B(\hat{P}(k))U(n) = B_{pu}(n)\hat{P}(k)$, $C(\hat{P}(k))x(n) = C_{px}(n)\hat{P}(k)$, $D(\hat{P}(k))U(n) = D_{pu}(n)\hat{P}(k)$. $P(k)$ is the real parameter, and $\hat{P}(k)$ is the estimated parameter within the k^{th} horizon window.

Following the above notations, $\epsilon(n+1)$ in the horizon window $kM \leq n \leq (k{+}1)M$ can be simplified to

$$\epsilon(n+1) = F(k)\epsilon(n) + \eta(k,n)e_p(k) = F\epsilon(n) + \eta(n)e_p. \tag{2.9}$$

Substitute (2.10) into the difference between the Lyapunov function at time $n+1$ and time n,

$$\begin{aligned} V(n+1){-}V(n) \quad &= \epsilon^T(n)F^TQF\epsilon(n) + e_p^T\eta^T(n)QF\epsilon(n) \\ &+\epsilon^T(n)F^TQ^T\eta(n)e_p + e_p^T\eta^T(n)Q\eta(n)e_p - \epsilon^T(n)Q\epsilon(n) \end{aligned}$$

Since $V(n{+}1){-}V(n)$ is a scalar, the cross product is a scalar, which results the relationship that $e_p^T\eta^T(n)QF\epsilon(n) = \epsilon^T(n)F^TQ^T\eta(n)e_p$. According to the Cauchy-Schwarz inequality, it results in the inequality

$$e_p^T\eta^T(n)QF\epsilon(n){+}\epsilon^T(n)F^TQ^T\eta(n)e_p{\leq}\omega e_p^T\eta^T(n)Q^TQ\eta(n)e_p{+}\frac{1}{\omega}\epsilon^T(n)F^TF\epsilon(n)$$

where ω is a positive scalar.

The above inequality is expressed as

$$\begin{aligned} V(n+1) - V(n) \leq \quad & \epsilon(n)^T[F^TQF - Q + \tfrac{1}{\omega}F^TF]\epsilon(n) \\ & +e_p^T[\eta^T(n)Q\eta(n) + \omega\eta^T(n)Q^TQ\eta(n)]e_p \end{aligned} \tag{2.10}$$

For the term of $e_p^T\eta^T(n)Q\eta(n)e_p$, there exists the following inequality that

$$\begin{aligned} e_p^T\eta^T(n)Q\eta(n)e_p{=}&||Q^{0.5}\eta(n)e_p||_2^2{\leq}||Q^{0.5}||_2^2||\eta(n)||_2^2||||e_p||_2^2 \\ &{\leq}\gamma||Q^{0.5}||_2^2||e_p||_2^2{\leq}\gamma||Q^{0.5}||_2^2\alpha||\epsilon(n)||_2^2{=}\alpha\gamma||Q^{0.5}||_2^2\epsilon^T(n)\epsilon(n) \end{aligned}$$

Similarly, there exists

$$\omega e_p^T\eta^T(n)Q^TQ\eta(n)e_p \leq \omega\gamma\alpha||Q||_2^2\epsilon^T(n)\epsilon(n).$$

Based on the above inequalities,

$$V(n+1)-V(n) \leq \epsilon^T(n)[F^TQF - Q + \frac{1}{\omega}F^TF + \alpha\gamma(||Q^{0.5}||_2^2 + \omega||Q||_2^2)]\epsilon(n)$$

The proper choices of positive scalars α, ω, γ, and the positive semi-definite matrix Q assure that

$$F^TQF - Q + \frac{1}{\omega}F^TF + \alpha\gamma(||Q^{0.5}||_2^2 + \omega||Q||_2^2) = -S$$

where S is the positive semi-definite matrix, therefore $V(n{+}1){-}V(n) \leq -\bar{\epsilon}^TS\bar{\epsilon} \leq 0$ is proved. That means the state estimation error is nonincreasing in one estimation window.

2.5 Simulation and Application in a Hydraulic Rig System

2.5.1 Modeling of Hydraulic Rig System

To validate this parameter and state estimation scheme, an industry example is simulated here. The hydraulic rig system consists of a stiff shaft driven by a hydraulic motor and loaded with a hydraulic pump and a servo valve which controls the oil flow to motor [30–32]. The dynamic equations of the system is shown as follows:

$$\begin{aligned} &K_vv(t) = T_1T_2\tfrac{dX_s^2(t)}{dt^2} + (T_1+T_2)\tfrac{dX_s(t)}{dt} + X_s(t) \\ &Q_v(t) = K_\theta X_s(t)(P_s(t)-P_m(t))^{0.5} \\ &Q_v(t) = C_rS_s(t) + \tfrac{V_t}{2\beta}\tfrac{dP_m(t)}{dt} + K_lP_m(t) \\ &T_m(t) = P_m(t)C_r\eta_m \\ &T_m(t) = I\tfrac{d\theta^2(t)}{dt^2} + D\tfrac{d\theta(t)}{dt} + P_p(t)C_r\eta_p \end{aligned}$$

where $X_s(t)$ is the displacement of the valve, $v(t)$ is the voltage input to the valve, $Q_v(t)$ is the flow rate through the valve, $P_s(t)$ is the supply pressure considered as constant here, $P_m(t)$ is the pressure differential across the motor, $S_s(t)$ is the shaft angular velocity which is equal to $d\theta/dt$ and $T_m(t)$ is the motor torque. All of the parameters in the above equations are listed in Table 2.1 [33]. The critical values are combined as a state variable $x = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}^T = \begin{bmatrix} P_m(t) & \dot{\theta}(t) & X_s(t) & \dot{X}_s(t) \end{bmatrix}^T$, here $n = 4$. The control variables are combined as an input vector $u = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^T = \begin{bmatrix} v(t) & P_p(t) \end{bmatrix}^T$. Outputs are $y_1 = x_1 + x_3$, $y_2 = 2x_3 + x_2$, $y_3 = x_3 - x_4$.

Table 2.1: PARAMETER VALUES IN A HYDRAULIC RIG SYSTEM

Parameter	Value	Description
$P_s(t)$	140	Supplied pressure
T_1	0.02	The eletro-magnetic time constant for the valve
T_2	0.01	The eletro-mechanic time constant for the value
V_t	0.01	Total trapped volume
K_s	-0.48	The valve eletro-magnetic gain
C_r	0.01	The motor displacement
β	3.30	Oil bulk modulus
K_t	0.15	Leakage coefficient
K_θ	2.4	The valve flow coefficient
η_m	0.95	Efficiency of the motor
η_p	0.89	Efficiency of the pump
I	$1.0 \cdot 10^{-5}$	Total inertia of pump, motor and shaft
D	$9.0 \cdot 10^{-5}$	Viscous friction coefficient

The system is reformulated in the following form,

$$x\dot{(t)} = f(x(t), u(t)) = \begin{bmatrix} -\frac{-2\beta K_l}{V_t} & -\frac{2\beta C_r}{V_t} & 0 & 0 \\ \frac{C_r \eta_m}{I} & -\frac{D}{I} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{1}{T_1 T_2} & -\frac{T_1+T_2}{T_1 T_2} \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0 \\ 0 & -\frac{C_r}{I\eta_p} \\ 0 & 0 \\ \frac{K_s}{T_1 T_2} & 0 \end{bmatrix} u(t) + \begin{bmatrix} \frac{2\beta K_\theta X_s}{V_t}(P_s - P_m)^{0.5} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Substituting all the parameter values, the system can be expressed as

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix} = \begin{bmatrix} -99 & -6.6 & 0 & 0 \\ 950 & -9 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -5000 & -150 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 \\ 0 & -1123.6 \\ 0 & 0 \\ -2400 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} 1584x_3(140 - x_1)^{0.5} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

It is given in [33] that the input signals are bounded in the range $u_1 \in [-1.6, -0.6]$ and $u_2 \in [34, 38]$, and the three operating points corresponding to $u_{o1}^{(1)} = -1.1, u_{o1}^{(2)} = 36$, $u_{o2}^{(1)} = -0.7, u_{o2}^{(2)} = 34.5$, $u_{o3}^{(1)} = -1.5, u_{o3}^{(2)} = 37.5$ are chosen, here $p = 3$. A nonlinear system can be linearized using Jacobian linearization around the operating points corresponding with the chosen input signals. The operating points are obtained by solving the equations $f(x_{oi}, u_{oi}) = 0$, $x_{o1} = \begin{bmatrix} 47.3959 \\ 508.5010 \\ 0.528 \\ 0 \end{bmatrix}$, $x_{o2} = \begin{bmatrix} 42.3378 \\ 161.8521 \\ 0.336 \\ 0 \end{bmatrix}$ and $x_{o3} = \begin{bmatrix} 52.2603 \\ 834.7029 \\ 0.72 \\ 0 \end{bmatrix}$.

Referring to Section 2.2, the discrete time local models can be gained from Equation (2.1) and (2.2) using zero-order-hold discretization with sampling time 0.1s. In the state estimation step $L(k)$ is designed such that the eigenvalues of the $Ap(k) + L(k)Cp(k)$ are at the origin at the end of each of parameter estimation window. One state estimation lasts for 2 seconds, and the sampling time is 0.1s, so the estimation window length M equals $2s\ 0.1s = 20$.

2.5.2 Result Testing and Case Study

In this section three simulation scenarios are built up using three kinds of operating input signals to validate the scheme.

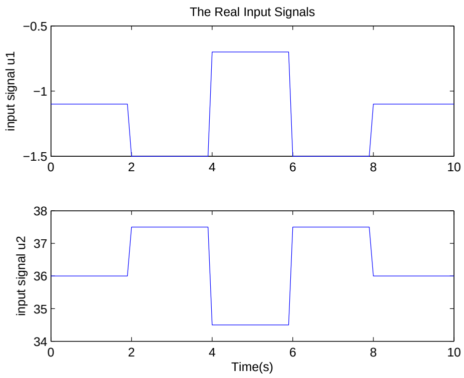


Figure 2.3: Input signals in case 1

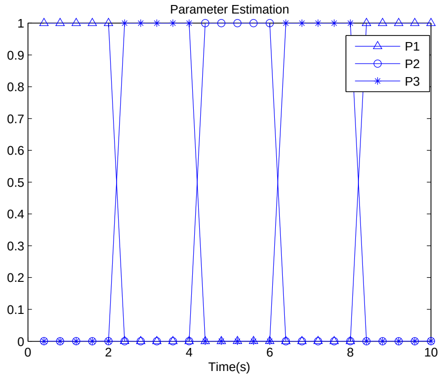


Figure 2.4: Estimated parameter in case 1

Case 1: Input That Matches Local Operating Input

In this case, the input signals are designed as square signals, whose values switch between the operating values of local models, as shown in Figure 2.3. The square wave inputs switch from one operating point of one local model to another in every two seconds. For example, the designed inputs of $u_1 = -1.1$ and $u_2 = 36$ implies that the nonlinear system can be approximated by the first linear local model during the first two seconds. Figure 2.4 reveals the estimated parameters in the first estimation window are $P_1(1) = 1$, $P_2(1) = 0$ and $P_3(1) = 0$. It means the first local model is selected perfectly in the first estimation window. Similarly, in the other time periods, the accurate parameter estimation, that one element of $P(k)$ is 1, validates the scheme when the nonlinear system operates at the operating points. The state variables of the continuous time real model show in Figure 2.5 and the estimated state variables show in Figure 2.6. There are some state transients between operating points' switchings in the real system, which results in the estimated errors between the sampled real states and estimated states have some outlier at the switching time instants, as shown show in Figure 2.7 and 2.8. However, the state errors finally become quite tiny in each estimation window.

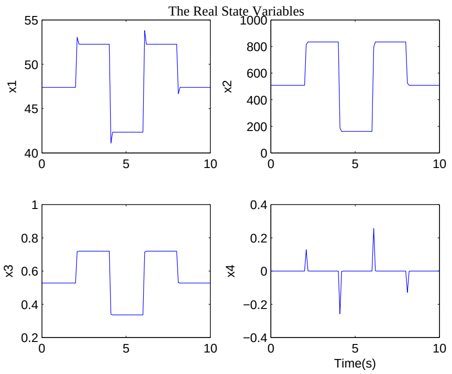


Figure 2.5: Real states in case 1

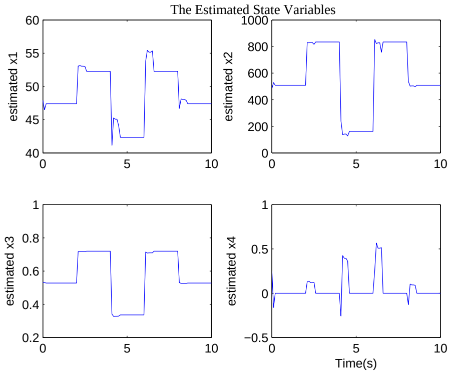


Figure 2.6: Estimated states in case 1

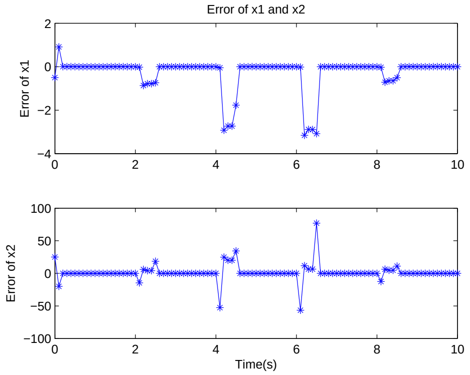


Figure 2.7: Est. errors of x_1, x_2 in case 1

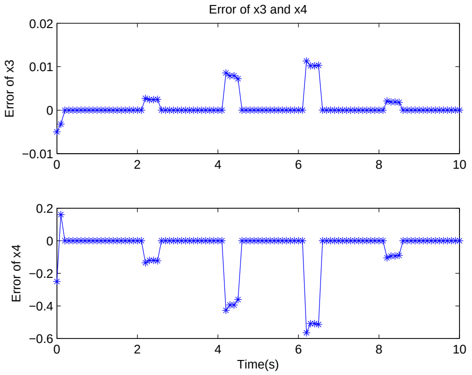


Figure 2.8: Est. errors of x_3, x_4 in case 1

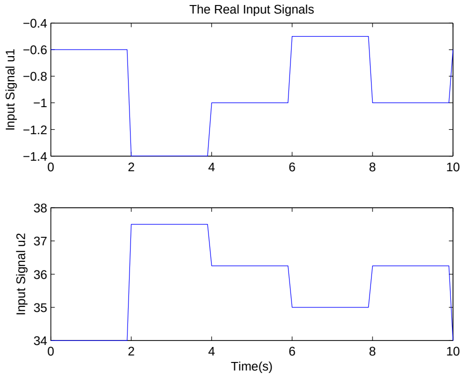


Figure 2.9: Input signals in case 2

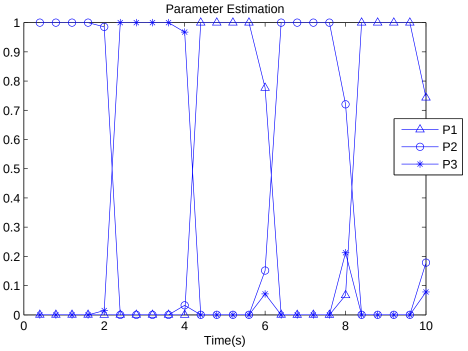


Figure 2.10: Estimated parameters in case 2

Case 2: The Square Wave Input Near Local Operating Point

As shown in Figure 2.9, the designed square wave inputs' amplitudes are near to the local inputs, and their values are $(-0.6, -1.4, -1, -0.5, -1)$ and $(34, 37.5, 36.25, 35, 36.25)$. In Figure 2.10, it illustrates that the estimated parameters are $P(5) = \begin{bmatrix}0.9855 & 0.0145 & 0\end{bmatrix}^T$, $P(10) = \begin{bmatrix}0 & 0.0327 & 0.9673\end{bmatrix}^T$, $P(15) = \begin{bmatrix}0.7767 & 0.1515 & 0.0718\end{bmatrix}^T$, $P(20) = \begin{bmatrix}0.0671 & 0.7203 & 0.2126\end{bmatrix}^T$, $P(25) = \begin{bmatrix}0.7432 & 0.1786 & 0.0782\end{bmatrix}^T$, they are very closed but not exactly the same as the parameter in case 1, $P = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$, $P = \begin{bmatrix}0 & 1 & 0\end{bmatrix}^T$, $P = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$. For the nonlinear system whose inputs are around the local inputs, the estimated parameters using this scheme are the values closed to $\begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$, $\begin{bmatrix}0 & 1 & 0\end{bmatrix}^T$ and $\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$.

In Figure 2.13 and 2.14, the estimated states follow the real states roughly. The errors between them become very small in each state estimation window.

Case 3: Damping Input Signals

A set of damping input signals $u_1(t) = -1.1 + 0.4e^{-0.5t}sin(\frac{\pi}{4}t)$, $u_2(t) = 36 + 1.5e^{-0.5t}sin(\frac{pi}{4}t)$ is designed in Case 3. As Figure 2.15 shows, the nonlinear system operates around the first local operating point at the first and final few time instants. The estimated parameters in case 3 are shown in Figure 2.16, where the parameter in the first and last few seconds is $\begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$. The trends of $P_1(k)$, $P_2(k)$

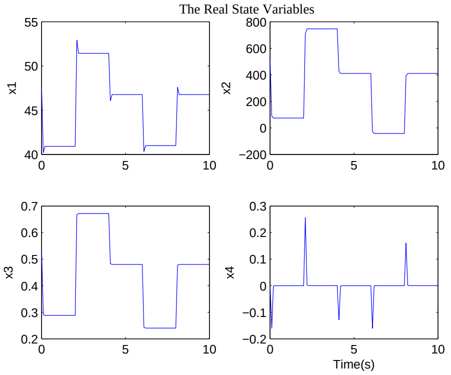


Figure 2.11: Real state variable in case 2

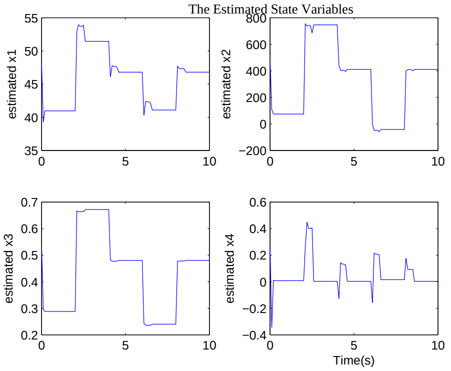


Figure 2.12: Estimated state in case 3

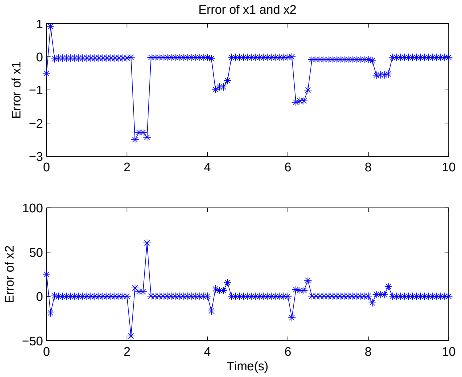


Figure 2.13: Est. errors of x_1, x_2 in case 2

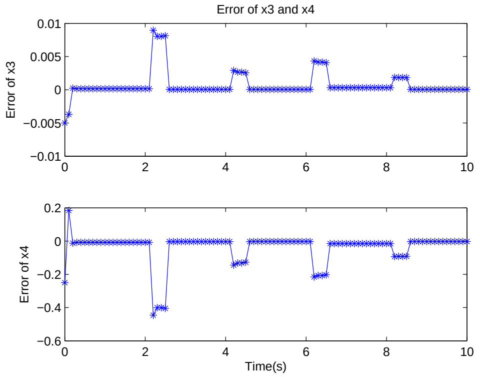


Figure 2.14: Est. errors of x_3, x_4 in case 2

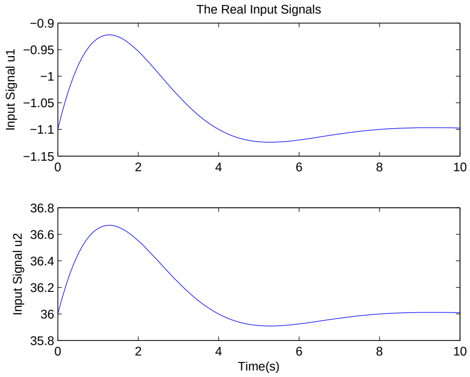


Figure 2.15: Input signals in case 3

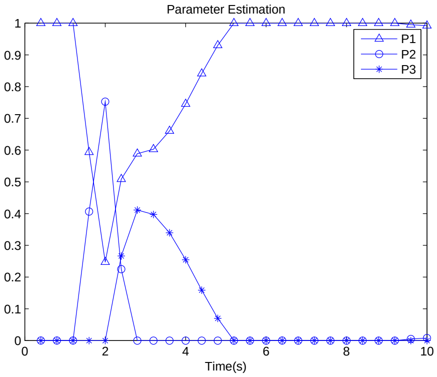


Figure 2.16: Estimated parameters in case 3

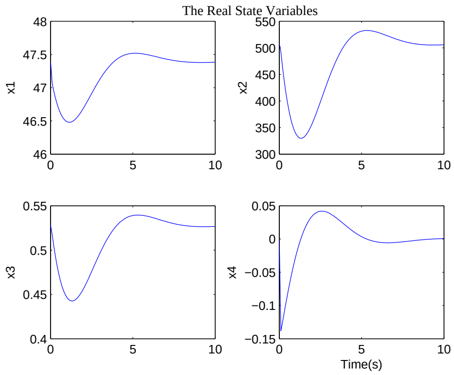


Figure 2.17: Real states in Case 3

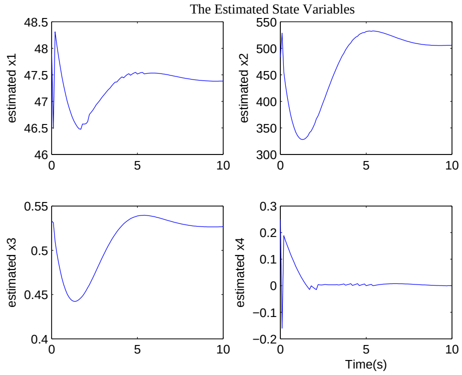


Figure 2.18: Estimated states in case 3

and $P_3(k)$ are consistent with the trend of the input signals.

The state values in a real system and the estimated state values show similar trends to the damping input signals. In addition, the estimated errors in Figure 2.19 and 2.20 decrease progressively to the extremely tiny values with time.

The three designed simulation scenarios with the three typical inputs show the fact that, for the nonlinear system operating around one of the linear local models, the corresponding parameter is estimated nearly as 1 and the parameters corresponding to other models are approximately 0. The estimated parameters can reflect the behaviors of a nonlinear system. The estimation errors are approaching zero within one estimation window, which validates the parameter and state estimation scheme again.

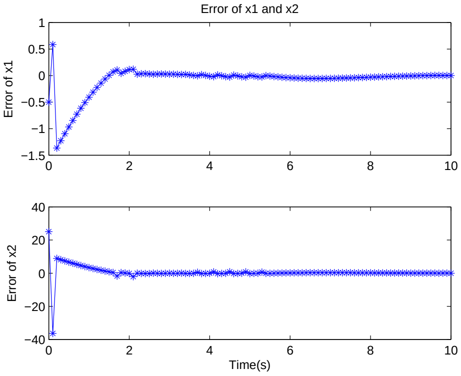


Figure 2.19: Est. errors of x_1, x_2 in case 3

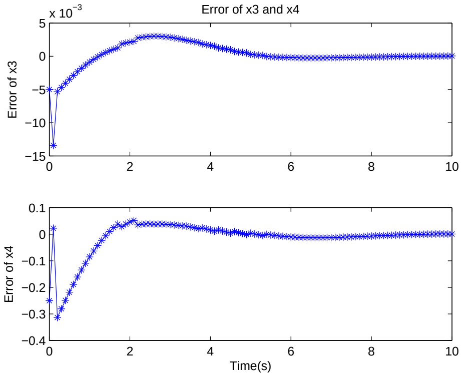


Figure 2.20: Est. errors of x_3, x_4 in case 3

Chapter 3

Application to Wind Turbine Systems

3.1 Introduction to wind turbine systems

3.1.1 Background of Wind Energy

Renewable energy provides humanity a clean and cost effective energy source to substitute for fossil fuels and nuclear power. Renewable and sustainable energy sources are necessary for future. Wind energy is one of the most abundant forms of renewable energy and could be harvested throughout the globe. It reduces dependency on fossil fuels, and decreases greenhouse gas emissions. As one of the fastest growing renewable-energy industries, world wind generation capacity quadrupled between 2000 and 2006. At the end of 2007, worldwide capacity for wind-powered generators was 94.1 GW, producing about one percent of the world's electricity. World Wind Energy Association records indicate that worldwide wind energy production by 2010 is expected to increase to 160 GW, which represents a yearly growth rate of 20 percent [34].

Wind energy offers many advantages, which explains why it has been the fast-growing energy source in the world. However, the development of wind power is also confronted with many technical challenges. Firstly, wind power has to compete with conventional power generation sources on a cost basis. Depending on the wind profile at the site, the wind farm may or may not be as cost competitive as a fossil fuel based power plant [34]. Wind energy cannot be stored, and not all winds can be harnessed to meet the timing of electricity demands. Furthermore, in the event of equipment failure, all grid-connected wind energy farms, for power regulation purposes, are required to have a guaranteed energy storage capacity that will allow for a consistent power transmission. Therefore the parameter and state estimations of a wind turbine system become vital in failure and fault detection.

3.1.2 Wind Turbine Model

The performance of the wind turbine is very sensitive to the wind speed. From a structural perspective, a wind turbine system consists of several subsystems such as a blade and pitch system, a drive train system, a generator and convertor system, as shown in Figure 3.1. According to the measured performance of the wind turbine system, the Quasi-LPV model and the linear fractional transformation model can be used to represent a wind turbine system [35]. Nevertheless the classical linearization around operating points is used in this work.

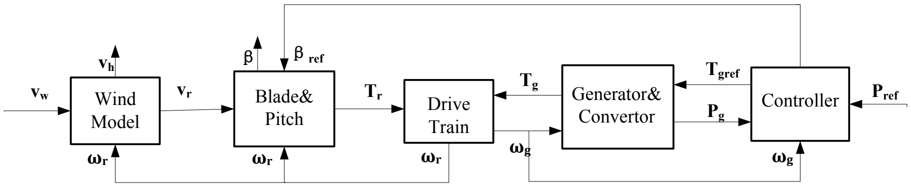


Figure 3.1: Relationship between subsystems in the wind turbine system

Blade and pitch subsystem

In the blade and pitch subsystem, as shown in Figure 3.1, the rotor effective wind speed v_r, pitch angle reference signal β_{ref} and rotor speed ω_r are inputs while pitch angle β and aerodynamic torque T_r are the outputs. The pitch actuator is represented as a second order model [36],

$$\frac{\beta(s)}{\beta_{ref}(s)} = \frac{e^{-t_d s}\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{3.1}$$

$$\ddot{\beta}(t) = -2\xi\omega_n\dot{\beta}(t) - \omega_n^2\beta(t) + \omega_n^2\beta_{ref}(t - t_d) \tag{3.2}$$

where t_d is the communication delay to the pitch actuator, ω_n is the natural frequency of the pitch actuator model and ξ is the damping ratio of the pitch actuator model.

The torque applied on the rotor is expressed as

$$T_r = \frac{1}{2\omega_r(t)} \rho A v_r^3(t) C_p(\lambda(t), \beta(t)) \tag{3.3}$$

where ρ is the air density, A is the rotor swept area, $C_p(\lambda(t), \beta(t))$ is the power coefficient which depends on pitch angle $\beta(t)$ and tip-speed ratio $\lambda(t)$. The tip-speed ratio is defined as the ratio between the tip speed of the blades and rotor effective wind speed:

$$\lambda(t) = \frac{\omega_r(t) R}{v_r(t)} \tag{3.4}$$

where R is the radius of the rotor.

Drive Train Subsystem

The aerodynamic torque T_r and generator torque T_g are the inputs, while pitch speed ω_r and generator speed ω_g are the outputs in the drive train subsystem. The drive train subsystem is used to transfer the aerodynamic torque to the generator for the purpose of increasing rotor speed.

For the low-speed shaft the dynamic equation [36] is given as

$$J_r \ddot{\theta}_r(t) = T_r(t) - T_l(t) - B_r \dot{\theta}_r(t)$$

where B_r is the viscous friction of the low-shaft; J_r is the moment of inertia of the low-speed shaft; $T_1(t)$ is the torque acting on the low-speed shaft; $\theta_r(t)$ is the angle of the low-speed shaft.

For the high-speed shaft, the dynamic equation is expressed as

$$J_g \ddot{\theta}_g(t) = T_h(t) - T_g(t) - B_r \dot{\theta}\ _g(t)$$

where B_g is the viscous friction of the high-speed shaft; J_g is the moment of inertia of the high-speed shaft; $T_g(t)$ is the generator torque; $T_h(t)$ is the torque acting on the high-speed shaft; $\theta_g(t)$ is the angle of the high-speed shaft.

The gearbox connects the low-speed shaft with the high-speed shaft, thus the torques acting on the two shafts have the relationship that

$$T_h(t) = \frac{T_l(t)}{N_g}$$

where N_g is the drive train ratio.

Considering the torsion spring and the friction coefficient, the torque of the low-speed shaft $T_l(t)$ and the torsion angle θ_Δ are given as follows,

$$T_l(t) = K_{dt}\theta_\Delta(t) + B_{dt}\dot{\theta}_\Delta(t)$$

$$\theta_\Delta(t) = \theta_r(t) - \frac{\theta_g(t)}{N_g}$$

where B_{dt} is the torsion damping coefficient of the drive train; K_{dt} is the torsion stiffness of the drive train; $\theta_\Delta(t)$ is the torsion angle of the drive train.

Overall all the equations above can be rewritten as

$$\begin{array}{l} J_r\dot{\omega}_r(t) = T_a(t) - K_{dt}\theta_\Delta(t) - (B_{dt} + B_r)\omega_r(t) + \frac{B_{dt}}{N_g}\omega_g(t) \\ J_g\dot{\omega}_g(t) = \frac{K_{dt}}{N_g}\theta_\Delta(t) + \frac{B_{dt}}{N_g}\omega_r(t) - (\frac{B_{dt}}{N_g} + B_g)\omega_g(t) - T_g(t) \\ \dot{\theta}_\Delta(t) = \omega_r(t) - \frac{1}{N_g}\omega_g(t) \end{array} \tag{3.5}$$

where $\omega_g(t) = \dot{\theta}_g(t)$ is the generator speed and $\omega_r(t) = \dot{\theta}_r(t)$ is the rotor speed.

Generator and Convertor Subsystem

In the generator and convertor subsystem the generator speed ω_g and the reference for generator torque $T_{g,ref}$ are inputs, while generator torque T_g and power produced by the generator P_g are outputs. The generator and convertor subsystem converts the mechanical energy into electrical energy.

According to the dynamic characteristics of the convertor, it can be approximated by a first order system with time delay as follows,

$$\frac{T_g(s)}{T_{g,ref}} = \frac{e^{-t_{g,d}}s}{\tau_g s + 1}$$

$$\dot{T}_g(t) = -\frac{1}{\tau_g}T_g(t) + \frac{1}{\tau_g}T_{g,ref}(t - t_{g,d}) \tag{3.6}$$

The produced power is determined by the rotor speed and generator torque, and it is given as:

$$P_g(t) = \eta_g\omega_g(t)T_g(t)$$

Controller Subsystem

As is illustrated in Figure 3.1, the controller subsystem has three inputs: the generator speed ω_g, the produced power P_g and the designed reference power P_{ref}. The reference pitch angle β_{ref} and the reference generator torque $T_{g,ref}$ are the outputs and are determined by the controller module.

In this thesis the control design is not discussed in detail, however a typical control strategy usually could be divided into three parts according to different ranges of wind velocity V_w, rotor speed ω_r and pitch angle β in order to maximize energy capture and minimize static loads, and to limit noise emission [37]. A typical control strategy of a wind turbine is illustrated in Figure 3.2.

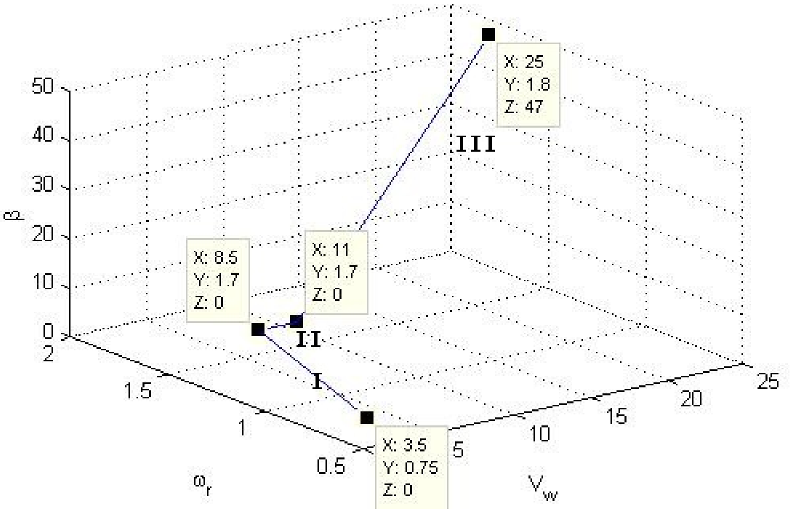


Figure 3.2: Operating locus of a typical wind turbine

It can be seen in Figure 3.2 that there are three subareas I, II and III:

- In subarea I, the control strategy is to keep the aerodynamic efficiency as high as possible. According to subarea I, the rotor speed ω_r increases proportionally with the wind velocity V_r, which means the generator speeds up along with the stronger wind to optimize the efficiency. On the other side, the pitch actuation does not take effect and pitch angle is kept maximum because of the relative low wind speed in this subarea.

- In subarea II the rotor speed of the wind turbine maintains constant and pitch actuation still does not take effect in this area, no matter if the wind speed increases or not.

- In subarea III, in order to reduce unstable fluctuations, the generated power maintains a reasonable constant due to the overrated wind. Therefore in this case the pitch actuation starts to work by adjusting the pitch angle and decreasing the rotor aerodynamic efficiency.

Since the wind turbine shows three different characteristics, three linearized models can describe the wind turbine system around the three operating points.

3.2 Wind Turbine Test Bench

The data for the wind turbine system in this thesis is obtained from a wind turbine test bench of kk-electronic a/s (refer to Appendix A for details), which was proposed by P. F. Odgaard [38]. A fault-free condition was assumed in this benchmark, which consists of six parts: sensors, control, blade & pitch, generator & converter, drive train and wind speed.

As discussed in Chapter 1, the main goal of this chapter is to build the wind turbine model which excludes the wind model and controlling system. To achieve this, the relationships between the variables in this simulation bench mark are re-constructed and summarized as follows:

Blade and Pitch Subsystem

The input wind velocity V_r, input rotor speed ω_r, input reference signal for pitch angle β_{ref} are the input variables from the wind model, the drive train subsystem, and the controller subsystem respectively. The outputs are rotor torque T_r, the three pitch angles β_1, β_2 and β_3.

According to the simulation relationship, we can obtain that,

$$\beta_1 = \beta_2 = \beta_3 = \frac{\omega_n^2}{s^2 + \xi\omega_n s + \omega_n^2}\beta_{ref}$$

$$T_r = C_p(\lambda, \beta) V_r^2 \rho \pi R^3$$

$$\lambda = \frac{\omega_r R}{V_r}$$

where $C_p(\lambda, \beta) = C_p(\frac{\omega_r R}{V_r}, \beta)$ is a nonlinear function with respect to ω_r, V_r and β. Since C_p is difficult to express in a linear or nonlinear equation, the relationship in the benchmark is given in a look-up table, as shown in Figure 3.3. The row breakpoints λ and the column breakpoints β are the two decision variables in the nonlinear function C_p.

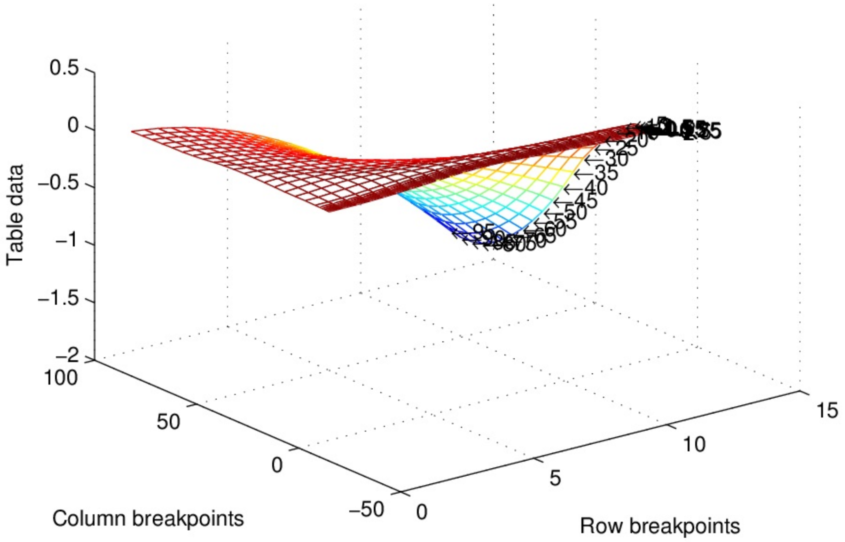


Figure 3.3: Nonlinearity of C_p respect to λ and β

The nonlinearity of the wind turbine is a result of nonlinearity on C_q, hence the linearization for the wind turbine model is achieved by utilizing the finite difference in the discrete data set.

Drive Train Subsystem

In the drive train subsystem the input rotor torque T_r is decided by the blade and pitch system, while the input generator torque T_g comes from the generator and convertor system. Outputs in this module are rotor speed ω_r and generator speed ω_g.

From the subsystem Simulink, a state space equation can be built up as

$$\begin{bmatrix}\dot{\omega}_r\\ \dot{\omega}_g\\ \dot{\theta}_\Delta\end{bmatrix} = A_{ddt}\begin{bmatrix}\omega_r\\ \omega_g\\ \theta_\Delta\end{bmatrix} + B_{ddt}\begin{bmatrix}T_r\\ T_g\end{bmatrix}$$

$$\begin{bmatrix}\omega_r\\ \omega_g\end{bmatrix} = C_{ddt}\begin{bmatrix}\omega_r\\ \omega_g\\ \theta_\Delta\end{bmatrix}$$

Reading from the variable data file, the system matrices in the above state space model are realized to $B_{ddt} = \begin{bmatrix}\frac{1}{J_r} & 0\\ 0 & -\frac{1}{J_g}\\ 0 & 0\end{bmatrix}$, $C_{ddt} = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$ and

$$A_{ddt} = \begin{bmatrix}-\frac{B_{dt}+B_r}{J_r} & \frac{B_{dt}}{N_g\cdot J_r} & -\frac{K_{dt}}{J_r}\\ \frac{\eta_{dt}B_{dt}}{N_g\cdot J_g} & -\frac{\eta_{dt}\cdot B_{dt}}{N_g^2\cdot J_g} - \frac{B_g}{J_g} & \frac{\eta_{dt}\cdot K_{dt}}{N_g\cdot J_g}\\ 1 & -\frac{1}{N_g} & 0\end{bmatrix}$$

Generator and Convertor Subsystem

In the generator & convertor subsystem the input signal, such as generator torque reference signal $T_{g,ref}$, is from the controller system, while the generator rate ω_g is an input signal is from the drive train module. The outputs are generator torque T_g and generated power P_g.

Based on the equation in the Simulink benchmark, there exists equations such that

$$T_g = \frac{\alpha_{gc}}{s+\alpha_{gc}}T_{g,ref}$$
$$P_g = \eta_{gc}T_g\omega_g$$

Referring to the variable data file, all the variable values are listed in Table 3.1. Therefore, the wind turbine model in this benchmark can be numerically realized by using these variable values.

3.3 Numerical Local Linear Models

The local models can be obtained by calculating the first derivative around the operating points, which is shown in Chapter 2. A detailed linearization process is

Table 3.1: VARIABLE VALUES IN THE BENCHMARK

T_s	R	H	k	α	ω_n	ξ	ρ	K_{dt}	$\tau_{g,d}$
0.01	57.5	87	4.7	0.1	11.11	0.6	1.225	2.7×10^9	0

B_{dt}	B_r	B_g	N_g	r_0	η_{dt}	J_g	J_r	α_{gc}	η_{gc}
0.7749	7.11	15.6	95	1.5	0.97	390	55×10^6	5×10^{-5}	0.98

not discussed here. Therefore, an assembled model in [36] has been used in this thesis, and summarized as follows,

$$
\begin{bmatrix} \dot{T}_g(t) \\ \dot{x}_t(t) \\ \ddot{x}_t(t) \\ \dot{\beta}(t) \\ \ddot{\beta}(t) \\ \dot{\theta}_\Delta(t) \\ \dot{\omega}_g(t) \\ \dot{\omega}_r(t) \end{bmatrix}
=
\begin{bmatrix}
-\alpha_{gc} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{K_t}{M_t} & a_{33} & a_34 & 0 & 0 & 0 & a_{38} \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & -\omega_n^2 & -2\xi\omega_n & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{N_g} & 1 \\
-\frac{1}{J_g} & 0 & 0 & 0 & 0 & \frac{K_{dt}}{J_g N_g} & a_{77} & \frac{B_{dt}}{J_g N_g} \\
0 & 0 & a_{83} & a_{84} & 0 & -\frac{K_{dt}}{J_r} & \frac{B_{dt}}{N_g J_r} & a_{88}
\end{bmatrix}
\begin{bmatrix} T_g(t) \\ x_t(t) \\ \dot{x}_t(t) \\ \beta(t) \\ \dot{\beta}(t) \\ \theta_\Delta(t) \\ \omega_g(t) \\ \omega_r(t) \end{bmatrix}
+
\begin{bmatrix}
0 & \alpha_{gc} & 0 \\
0 & 0 & 0 \\
b_{31} & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & \omega_n^2 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
b_{81} & 0 & 0
\end{bmatrix}
\begin{bmatrix} v_r(t) \\ T_{g,ref}(t) \\ \beta_{ref}(t) \end{bmatrix}
\tag{3.7}
$$

where $x_t(t)$ is the displacement of the nacelle from its equilibrium position.

The wind speed, acting on the rotor, may cause a thrust which makes the wind tower sway back and forth, but in the benchmark the tower influence is negligible so $\dot{x}_t(t) \equiv 0$ and $\ddot{x}_t(t) \equiv 0$. Therefore the assembled model in the benchmark can be simplified as

$$
\begin{aligned}
\begin{bmatrix} \dot{T}_g(t) \\ \dot{\beta}(t) \\ \ddot{\beta}(t) \\ \dot{\theta}_\Delta(t) \\ \dot{\omega}_g(t) \\ \dot{\omega}_r(t) \end{bmatrix} &= \begin{bmatrix} -\alpha_{gc} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -\omega_n^2 & -2\xi\omega_n & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{N_g} & 1 \\ -\frac{1}{J_g} & 0 & 0 & \frac{K_{dt}}{J_g N_g} & a_{77} & \frac{B_{dt}}{J_g N_g} \\ 0 & a_{84} & 0 & -\frac{K_{dt}}{J_r} & \frac{B_{dt}}{N_g J_r} & a_{88} \end{bmatrix} \begin{bmatrix} T_g(t) \\ \beta(t) \\ \dot{\beta}(t) \\ \theta_\Delta(t) \\ \omega_g(t) \\ \omega_r(t) \end{bmatrix} \\
&+ \begin{bmatrix} 0 & \alpha_{gc} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \omega_n^2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_{81} & 0 & 0 \end{bmatrix} \begin{bmatrix} v_r(t) \\ T_{g,ref}(t) \\ \beta_{ref}(t) \end{bmatrix}
\end{aligned} \tag{3.8}
$$

where

$$
\begin{aligned}
a_{77} &= -\left(\frac{\eta_{dt} B_{dt}}{J_g N_g^2} + \frac{B_g}{J_g}\right) & a_{84} &= \frac{1}{3J_r}\frac{\partial T_r}{\partial \beta}\Big|_{\substack{\beta(t)=\bar{\beta} \\ \omega_r(t)=\bar{\omega}_r \\ v_r(t)=\bar{v}_r}} \\
a_{88} &= -\frac{B_{dt}+B_r}{J_r} + \frac{1}{J_r}\frac{\partial T_r(t)}{\partial \omega_r}\Big|_{\substack{\beta(t)=\bar{\beta} \\ \omega_r(t)\bar{\omega}_r \\ v_r(t)=\bar{v}_r}} & b_{81} &= \frac{1}{3J_r}\frac{\partial T_r(t)}{\partial v_r}\Big|_{\substack{\beta(t)=\bar{\beta} \\ \omega_r(t)=\bar{\omega}_r \\ v_r(t)=\bar{v}_r}}
\end{aligned}
$$

Based on the expression of T_r in Equation (3.3), the first derivative of T_r is related to the first derivative of C_p. In the benchmark C_p is obtained by a lookup table with coordinates λ and β and can be shown in three dimension mesh figure, as illustrated in Figure 3.3. As different operating points $(\bar{\beta}, \bar{\omega}_r, \bar{v}_r)$ are chosen, the linear local models vary in the elements a_{84}, a_{88} and b_{81} of the system matrices A and B.

According to the actual operating locus as shown in Figure 3.4, which is plotted using the data of the benchmark, and the actual operating locus's axonometric projection in Figure 3.5, Figure 3.6 and Figure 3.7, the nonlinear wind turbine model can be concluded as the composition of three linear local models in each range of (v_r, β, ω_r),

- Model 1 $v_r \in [3.5, 8.5]$, $\beta = 0$, $\omega_r \in [0.75, 1.6]$;

- Model 2 $v_r \in [8.5, 11]$, $\beta = 0$, $\omega_r = 1.65$;

- Model 3 $v_r \in [11, \infty)$, $\beta \neq 0$, $\omega_r \in [1.65, 1.8]$.

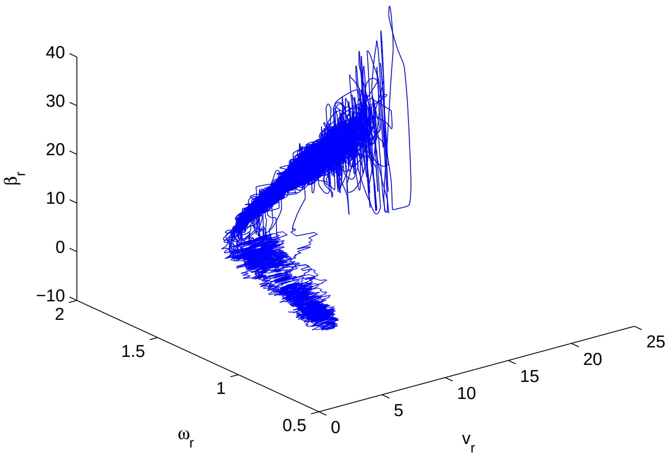


Figure 3.4: Actual operating locus

Three operating points, such as ($\bar{v}_r^{(1)} = 6$, $\bar{\beta}^{(1)} = 0$, $\bar{\omega}_r^{(1)} = 0.954$), ($\bar{v}_r^{(2)} = 10$, $\bar{\beta}^{(2)} = -0.5248$, $\bar{\omega}_r^{(2)} = 1.7047$), ($\bar{v}_r^{(3)} = 20$, $\bar{\beta}^{(3)} = 20.0516$, $\bar{\omega}_r^{(3)} = 1.7186$), are picked from each range. Around these operating points, $\frac{\partial T_r}{\partial \beta}$, $\frac{\partial T_r}{\partial \omega_r}$ and $\frac{\partial T_r}{\partial v_r}$ can be developed as the values below, and substituting $\rho = 1.225$, $R = 57.5$ in Table 3.1.

- For the operating point ($\bar{v}_r^{(1)} = 6, \bar{\beta}^{(1)} = 0, \bar{\omega}_r^{(1)} = 0.954$), the decision variable $\bar{\lambda}^{(1)}$ in the nonlinear function C_p is calculated by $\bar{\lambda}^{(1)} = \frac{\omega_r^{(1)} R}{v_r^{(1)}} = \frac{0.954 \times 57.5}{6} = 9.144$. The first derivative of $T_r(t)$ with respect to $\beta(t)$ is calculated by

$$\frac{\partial T_r(t)}{\partial \beta(t)} = \frac{\partial \frac{1}{2\omega_r(t)} \rho A v_r^3(t) C_p(\lambda(t), \beta(t))}{\partial \beta(t)} = \frac{\rho A v_r^3(t)}{2\omega_r(t)} \frac{\partial C_p(\lambda(t), \beta(t))}{\partial \beta(t)} \Big|_{\substack{v_r(t)=6 \\ \beta(t)=0 \\ \omega_r(t)=0.954 \\ \lambda(t)=9.144}}$$

$$= \frac{6.3587 \times 10^3 v_r^3(t)}{\omega_r(t)} \frac{\partial C_p(\lambda(t), \beta(t))}{\partial \beta(t)} \Big|_{\substack{v_r(t)=6 \\ \beta(t)=0 \\ \omega_r(t)=0.954 \\ \lambda(t)=9.144}}$$

According to the C_p lookup table, the first derivative of C_p to β can be calculated by the finite difference method:

$$\frac{\partial C_p(\lambda(t), \beta(t))}{\partial \beta(t)} \Big|_{\beta(t)=0, \lambda=9.144} = \frac{0.0504 - 0.0486}{0.5} = 0.0036$$

Therefore around the first operating point, the first derivative of $T_r(t)$ with

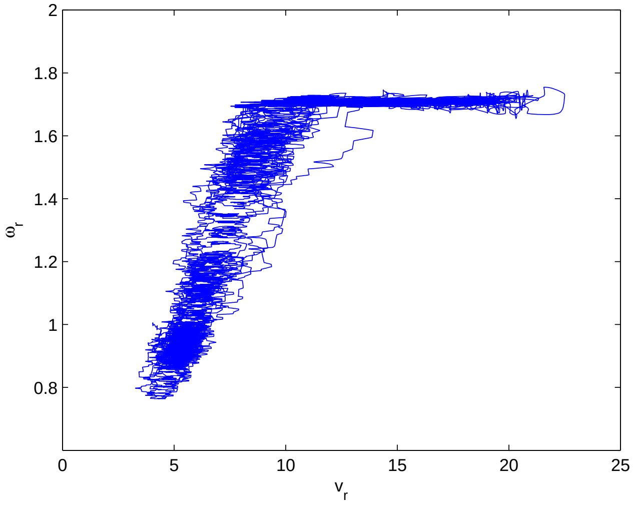


Figure 3.5: Axonometric projection of ω_r and v_r

respect to $\beta(t)$ is valued as

$$\frac{\partial T_r(t)}{\partial \beta(t)}\Big|_{\substack{v_r(t)=6\\ \beta(t)=0\\ \omega_r(t)=0.954\\ \lambda(t)=9.144}} = 5.1819\times 10^3 \tag{3.9}$$

Similarly, the first derivative of $T_r(t)$ with respect to $v_r(t)$ is calculated by

$$\frac{\partial T_r(t)}{\partial v_r(t)} = \frac{\partial \frac{1}{2\omega_r(t)}\rho A v_r^3(t) C_p(\lambda(t),\beta(t))}{\partial v_r(t)}$$

$$= \left(\frac{3\rho A v_r^2 C_p(\lambda(t),\beta(t))}{2\omega_r} + \frac{\rho A v_r^3(t)}{2\omega_r(t)}\frac{\partial C_p(\frac{\omega_r(t)R}{v_r(t)},\beta(t))}{\partial v_r(t)}\right)\Big|_{\substack{v_r(t)=6\\ \beta(t)=0\\ \omega_r(t)=0.954\\ \lambda(t)=9.144}}$$

$$= \left(\frac{3\rho A v_r^2 C_p(\lambda(t),\beta(t))}{2\omega_r} + \frac{\rho A v_r^3(t)}{2\omega_r(t)}\frac{\partial C_p(\frac{\omega_r(t)R}{v_r(t)},\beta(t))}{\partial \lambda(t)}(-\omega_r(t)Rv_r(t)^{-2})\right)\Big|_{\substack{v_r(t)=6\\ \beta(t)=0\\ \omega_r(t)=0.954\\ \lambda(t)=9.144}}$$

$$= \left(\frac{1.9076\times 10^4 v_r(t)^2 C_p(\lambda(t),\beta(t))}{\omega_r(t)} - \frac{3.6563\times 10^5 v_r(t)\partial C_p(\lambda(t),\beta(t))}{\partial \lambda(t)}\right)_{\substack{v_r(t)=6\\ \beta(t)=0\\ \omega_r(t)=0.954\\ \lambda(t)=9.144}}$$

Referring to the lookup table, $C_p(\lambda(t),\beta(t))|_{\lambda(t)=9.144,\beta(t)=0} = 0.0495$ and $\frac{\partial C_p(\lambda(t),\beta(t))}{\partial \lambda(t)} = |_{\lambda(t)=9.144,\beta(t)=0} = -0.0081$ can be substituted in the above equation to get the first derivative of $T_r(t)$ with respect to $v_r(t)$, expressed as

$$\frac{\partial T_r(t)}{\partial v_r(t)}\Big|_{\substack{v_r(t)=6\\ \beta(t)=0\\ \omega_r(t)=0.954\\ \lambda(t)=9.144}} = 5.3396\times 10^4 \tag{3.10}$$

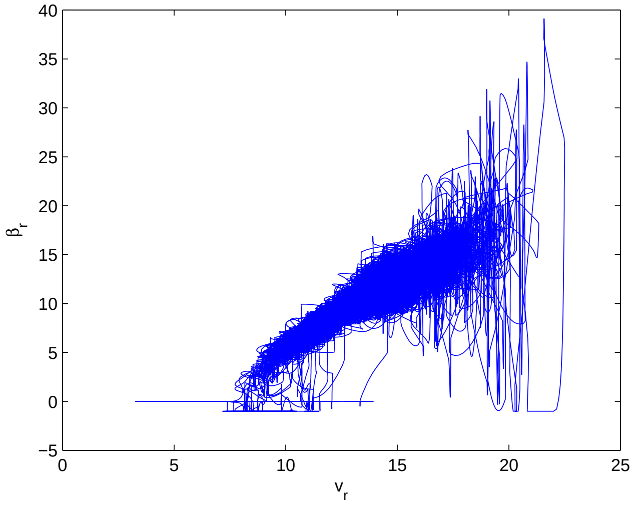


Figure 3.6: Axonometric projection of v_r and β

Similarly, the first derivative of $T_r(t)$ with respect to $\omega_r(t)$ is calculated by

$$\begin{aligned}
\frac{\partial T_r(t)}{\partial \omega_r(t)} &= \frac{\partial \frac{1}{2\omega_r(t)} \rho A v_r^3(t) C_p(\lambda(t), \beta(t))}{\partial \omega_r(t)} \\
&= \left(-\frac{\rho A v_r^3(t) C_p(\lambda(t), \beta(t))}{2\omega_r^2(t)} + \frac{\rho A v_r^3(t)}{2\omega_r(t)} \frac{\partial C_p(\frac{\omega_r(t)R}{v_r(t)}, \beta(t))}{\partial \omega_r(t)} \right) |_{\substack{v_r(t)=6 \\ \beta(t)=0 \\ \omega_r(t)=0.954 \\ \lambda(t)=9.144}} \\
&= \left(\frac{\rho A v_r^3(t) C_p(\lambda(t), \beta(t))}{2\omega_r^2(t)} + \frac{\rho A v_r^3(t)}{2\omega_r(t)} \frac{\partial C_p(\frac{\omega_r(t)R}{v_r(t)}, \beta(t))}{\partial \lambda(t)} (\frac{R}{v_r(t)}) \right) |_{\substack{v_r(t)=6 \\ \beta(t)=0 \\ \omega_r(t)=0.954 \\ \lambda(t)=9.144}} \\
&= \left(\frac{-6.3581\times 10^3 v_r^3(t) C_p(\lambda(t), \beta(t))}{\omega_r^2(t)} + \frac{3.6563\times 10^5 v_r^2(t)}{\omega_r(t)} \frac{\partial C_p(\lambda(t), \beta(t))}{\partial \lambda(t)} \right) |_{\substack{v_r(t)=6 \\ \beta(t)=0 \\ \omega_r(t)=0.954 \\ \lambda(t)=9.144}}
\end{aligned}$$

Referring to the lookup table, $C_p(\lambda(t), \beta(t))|_{\lambda(t)=9.144, \beta(t)=0} = 0.0495$ and $\frac{\partial C_p(\lambda(t),\beta(t))}{\partial \lambda(t)} = |_{\lambda(t)=9.144, \beta(t)=0} = -0.0081$ can be substituted in the above equation to get

$$\frac{\partial T_r(t)}{\partial v_r(t)} |_{\substack{v_r(t)=6 \\ \beta(t)=0 \\ \omega_r(t)=0.954 \\ \lambda(t)=9.144}} = -1.864 \times 10^5 \tag{3.11}$$

- For the operating point $(\bar{v}_r^{(2)} = 10, \bar{\beta}^{(2)} = -0.5248, \bar{\omega}_r^{(2)} = 1.7047)$, the corresponding variable is $\bar{\lambda}^{(2)} = \frac{\omega_r^{(1)} R}{v_r^{(1)}} = \frac{1.7047 \times 57.5}{10} = 9.8$.

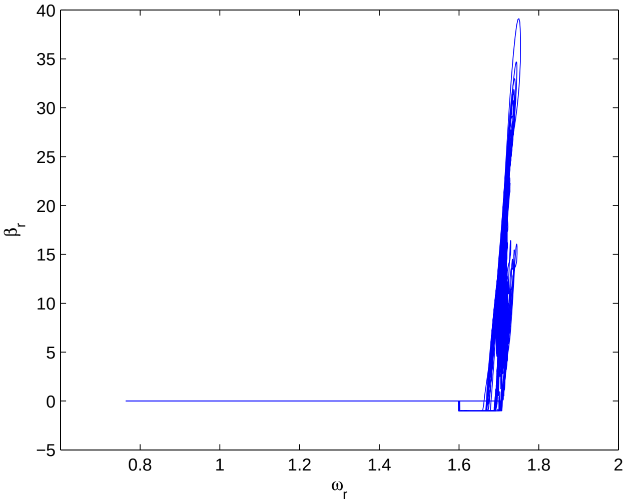


Figure 3.7: Axonometric projection of ω_r and β

Using the same method with the first operating point, the first derivative of $C_p(t)$ with respect to $\beta(t)$ and $\lambda(t)$ are respectively as follows,

$$\frac{\partial C_p(\lambda(t),\beta(t))}{\partial \beta(t)}|_{\beta(t)=-0.5248,\lambda=9.8} = \frac{0.0369-0.0342}{0.25} = 0.0108$$

$$C_p(\lambda(t),\beta(t))|_{\lambda(t)=9.8,\beta(t)=-0.5248} = 0.0369$$

$$\frac{\partial C_p(\lambda(t),\beta(t))}{\partial \lambda(t)} = |_{\lambda(t)=9.8,\beta(t)=-0.5248} = \frac{0.0369-0.0432}{0.5} = -0.0126$$

Therefore the first derivatives of $T_r(t)$ with respect to $\beta(t)$, $v_r(t)$ and $\omega_r(t)$ around the second operating point are calculated to

$$\frac{\partial T_r(t)}{\partial \beta(t)}|_{\substack{v_r(t)=10\\ \beta(t)=-0.5248\\ \omega_r(t)=1.7047\\ \lambda(t)=9.8}} = 4.03\times 10^4 \tag{3.12}$$

$$\frac{\partial T_r(t)}{\partial v_r(t)}|_{\substack{v_r(t)=10\\ \beta(t)=-0.5248\\ \omega_r(t)=1.7047\\ \lambda(t)=9.8}} = 8.736\times 10^4 \tag{3.13}$$

$$\frac{\partial T_r(t)}{\partial v_r(t)}|_{\substack{v_r(t)=10\\ \beta(t)=-0.5248\\ \omega_r(t)=1.7047\\ \lambda(t)=9.8}} = -3.51\times 10^5 \tag{3.14}$$

- For the operating point ($\bar{v}_r^{(3)} = 20, \bar{\beta}^{(3)} = 20.0516, \bar{\omega}_r^{(3)} = 1.7186$), the corresponding variable is $\bar{\lambda}^{(3)} = \frac{\omega_r^{(3)}R}{v_r^{(3)}} = \frac{1.7186\times57.5}{20} = 4.94$.

  Using the same method with the first and second operating points, the first derivatives of $C_p(t)$ with respect to $\beta(t)$ and $\lambda(t)$ around the third operating point are respectively as follows,

$$\frac{\partial C_p(\lambda(t),\beta(t))}{\partial \beta(t)}|_{\beta(t)=20.0516,\lambda=4.94} = \frac{-0.0378-0.0198}{10} = -0.00581$$

$$\frac{\partial C_p(\lambda(t),\beta(t))}{\partial \lambda(t)} = |_{\lambda(t)=4.94,\beta(t)=20.0516} = -0.0207$$

  Therefore the first derivatives of $T_r(t)$ with respect to $\beta(t)$, $v_r(t)$ and $\omega_r(t)$ around the third operating point are calculated to

$$\frac{\partial T_r(t)}{\partial \beta(t)}|_{\substack{v_r(t)=20\\ \beta(t)=20.0516\\ \omega_r(t)=1.7186\\ \lambda(t)=4.94}} = -1.7197\times10^5 \tag{3.15}$$

$$\frac{\partial T_r(t)}{\partial v_r(t)}|_{\substack{v_r(t)=20\\ \beta(t)=20.0516\\ \omega_r(t)=1.7186\\ \lambda(t)=4.94}} = 11.9404\times10^4 \tag{3.16}$$

$$\frac{\partial T_r(t)}{\partial v_r(t)}|_{\substack{v_r(t)=20\\ \beta(t)=20.0516\\ \omega_r(t)=1.7186\\ \lambda(t)=4.94}} = -1.63656\times10^6 \tag{3.17}$$

The first derivatives $\frac{\partial T_r(t)}{\partial \beta(t)}$, $\frac{\partial T_r(t)}{\partial v_r(t)}$ and $\frac{\partial T_r(t)}{\partial \omega_r(t)}$ have been obtained from Equations (3.9)-(3.17). As soon as they are substituted into the a_{84}, a_{88} and b_{81} in Equation (3.8), the local linear models are generated around the three operating points.

- For the first local model, $a_{84}^{(1)} = 3.14\times10^{-5}$, $a_{88}^{(1)} = -3.389\times10^{-3}$, $b_{81}^{(1)} = 3.3258\times10^{-4}$.

- For the second local model, $a_{84}^{(2)} = 2.44\times10^{-4}$, $a_{88}^{(2)} = -6.382\times10^{-3}$, $b_{81}^{(2)} = 5.294\times10^{-4}$.

- For the third local model, $a_{84}^{(3)} = 1.042\times10^{-3}$, $a_{88}^{(3)} = 2.9756\times10^{-2}$, $b_{81}^{(3)} = 7.2359\times10^{-4}$.

The numerical local models can be realized by substituting a_{84}, a_{88}, b_{81} and all the values in Table 3.1 into Equation (3.7). The system matrices of the local models are (A_1, B_1), (A_2, B_2) and (A_3, B_3), which are represented as

$$A_1 = \begin{bmatrix} -50 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -123.4321 & -13.332 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{95} & 1 \\ -\frac{1}{390} & 0 & 0 & 72874.5 & -0.04 & 2.09 \times 10^{-5} \\ 0 & 3.14 \times 10^{-5} & 0 & -49.09 & 3.61 \times 10^{-11} & -3.389 \times 10^{-3} \end{bmatrix}$$

$$A_2 = \begin{bmatrix} -50 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -123.4321 & -13.332 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{95} & 1 \\ -\frac{1}{390} & 0 & 0 & 72874.5 & -0.04 & 2.09 \times 10^{-5} \\ 0 & 2.44 \times 10^{-4} & 0 & -49.09 & 3.61 \times 10^{-11} & -6.382 \times 10^{-3} \end{bmatrix}$$

$$A_3 = \begin{bmatrix} -50 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -123.4321 & -13.332 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{95} & 1 \\ -\frac{1}{390} & 0 & 0 & 72874.5 & -0.04 & 2.09 \times 10^{-5} \\ 0 & 1.042 \times 10^{-3} & 0 & -49.09 & 3.61 \times 10^{-11} & 2.9756 \times 10^{-2} \end{bmatrix}$$

$$B_1 = \begin{bmatrix} 0 & 50 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 123.4321 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 3.2358 \times 10^{-4} & 0 & 0 \end{bmatrix} B_2 = \begin{bmatrix} 0 & 50 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 123.4321 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 5.294 \times 10^{-4} & 0 & 0 \end{bmatrix}$$

$$B_3 = \begin{bmatrix} 0 & 50 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 123.4321 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 7.2359 \times 10^{-4} & 0 & 0 \end{bmatrix}$$

The fact that system matrices of the three local models are ill-conditioned matrices brings losses of the vital information in the system during data processing. In order to balance the ill-conditioned matrix, a similarity transformation T is applied here and the similar model is presented as

$$\begin{aligned}\dot{\bar{x}}(t) &= TAT^{-1}\bar{x}(t) + TBu(t) = \bar{A}\bar{x}(t) + \bar{B}u(t)\\ y(t) &= CT^{-1}\bar{x}(t) + Du(t) = \bar{C}\bar{x} + \bar{D}u(t)\end{aligned} \tag{3.18}$$

where $\bar{x} = Tx$, $\bar{A} = TAT^{-1}$, $\bar{B} = TB$, $\bar{C} = CT^{-1}$ and $\bar{D} = D$. Here the similarity transformation matrix T can be chosen as

$$T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 16 \\ 0 & 8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 128 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0625 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Then the system matrices in the similar model are transferred as

$$\bar{A}_1 = \begin{bmatrix} 0.9989 & 0 & 0 & -0.0606 & 0.0066 & 0 \\ 0 & 0.9941 & 0.0747 & 0 & 0 & 0 \\ 0 & -0.1442 & 0.8695 & 0 & 0 & 0 \\ 0.079 & 0 & 0 & 0.9595 & -0.2126 & 0 \\ 0.0141 & 0 & 0 & 0.3509 & 0.9615 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6065 \end{bmatrix}$$

$$\bar{A}_2 = \begin{bmatrix} 0.9996 & 0 & 0 & -0.0606 & 0.0066 & 0 \\ 0 & 0.9941 & 0.0747 & 0 & 0 & 0 \\ 0 & -0.1442 & 0.8695 & 0 & 0 & 0 \\ 0.079 & 0 & 0 & 0.9595 & -0.2126 & 0 \\ 0.0141 & 0 & 0 & 0.3509 & 0.9615 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6065 \end{bmatrix}$$

$$\bar{A}_3 = \begin{bmatrix} 1.0053 & -0.002 & -0.0001 & -0.0608 & 0.0066 & 0 \\ 0 & 0.9941 & 0.0747 & 0 & 0 & 0 \\ 0 & -0.1442 & 0.8695 & 0 & 0 & 0 \\ 0.0792 & -0.0001 & 0 & 0.9595 & -0.2126 & 0 \\ 0.0142 & 0 & 0 & 0.3509 & 0.9615 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6065 \end{bmatrix}$$

$$\bar{B}_1 = \begin{bmatrix} -0.0198 & 0 & 0 \\ 0 & 0 & 0.0471 \\ 0 & 0 & 1.1511 \\ -0.0008 & 0 & 0 \\ -0.0001 & 0 & 0 \\ 0 & 0.3935 & 0 \end{bmatrix} \bar{B}_2 = \begin{bmatrix} -0.033 & 50 & 0 \\ 0 & 0 & 0.0471 \\ 0 & 0 & 1.1511 \\ -0.0013 & 0 & 0 \\ -0.0002 & 0 & 0 \\ 0 & 0.3935 & 0 \end{bmatrix}$$

$$\bar{B}_3 = \begin{bmatrix} -0.5035 & 0 & 0 \\ 0 & 0 & 0.0471 \\ 0 & 0 & 1.1511 \\ -0.02 & 0 & 0 \\ -0.0024 & 0 & 0 \\ 0 & 0.3935 & 0 \end{bmatrix}$$

3.4 Parameter under Discretization

The state and parameter estimation is carried out in a discrete time model. The zero-order hold discretization is utilized after the similarity transformation. It results that the parameter $\hat{P}(k)$ estimated from the scheme is not the estimation value of the real original parameter $P(k)$.

Based on the balanced models equation (3.18) in Section 3.3, the discrete balanced local models using zero-order hold can be expressed as

$$\begin{aligned} \bar{x}(k+1) &= e^{TA_iT^{-1}T_s}\bar{x}(k) + (TA_iT^{-1})^{-1}(e^{TA_iT^{-1}T_s} - I)TB_iu(k) \\ y(k) &= C_iT^{-1}\bar{x}(k) + D_iu(k) \end{aligned} \tag{3.19}$$

where i is the index of the specific local model, T_s is the sampling time. The LPV model of the wind turbine system is the linear combination of the above local models and the estimated parameter $\hat{P}(k)$, as shown below,

$$\begin{aligned} \bar{x}(k{+}1) &= \sum_{i=1}^{3} \hat{P}_i(k)e^{TA_iT^{-1}T_s}\bar{x}(k) + \sum_{i=1}^{3} \hat{P}_i(k)(TA_iT^{-1})^{-1}(e^{TA_iT^{-1}T_s} - I)TB_iu(k) \\ y(k) &= \sum_{i=1}^{3} \hat{P}_i(k)C_iT^{-1}\bar{x}(k) + \sum_{i=1}^{3} \hat{P}_i(k)D_iu(k) \end{aligned} \tag{3.20}$$

However, in the scheme a linear combination expression is expected to approximate the real nonlinear system, which has the form

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{3} P_i(k)A_ix(t) + \sum_{i=1}^{3} P_i(k)B_iu(t) \\ y(t) &= \sum_{i=1}^{3} P_i(k)C_ix(t) + \sum_{i=1}^{3} P_i(k)D_iu(t) \end{aligned} \tag{3.21}$$

where $k \cdot M \cdot T_s \leq t \leq (k+1) \cdot M \cdot T_s$. The real system is discretized using zero-order hold as

$$\begin{aligned} x(k{+}1) &= e^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} x(k) + (\sum_{i=1}^{3} P_i(k)A_i)^{-1}(e^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} - I)(\sum_{i=1}^{3} P_i(k)B_i)u(k) \\ y(k) &= \sum_{i=1}^{3} P_i(k)C_ix(k) + \sum_{i=1}^{3} P_i(k)D_iu(k) \end{aligned} \tag{3.22}$$

The similarity transformation is applied to the discrete time real system,

$$\begin{aligned} \bar{x}(k{+}1) &= Te^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} T^{-1}\bar{x}(k) + \\ &\quad T(\sum_{i=1}^{3} P_i(k)A_i)^{-1}(e^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} - I)(\sum_{i=1}^{3} P_i(k)B_i)u(k) \\ y(k) &= \sum_{i=1}^{3} P_i(k)C_iT^{-1}\bar{x}(k) + \sum_{i=1}^{3} P_i(k)D_iu(k) \end{aligned} \tag{3.23}$$

In order to make the model of the estimating scheme in Equation (3.20) approach the real model in Equation (3.23), the model of the estimation scheme is forced to equal the real model, and the corresponding system matrices are equivalent to

$$\begin{aligned} Te^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} T^{-1} &= \sum_{i=1}^{3} \hat{P}_i(k) e^{TA_iT^{-1}T_s} \\ T(\sum_{i=1}^{3} P_i(k)A_i)^{-1}(e^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} - I)(\sum_{i=1}^{3} P_i(k)B_i) & \\ &= \sum_{i=1}^{3} \hat{P}_i(k)(TA_iT^{-1})^{-1}(e^{TA_iT^{-1}T_s} - I)TB_i \end{aligned}$$

A_{d0} and B_{d0} are defined as the discrete time system matrices of the real model using zero-order hold discretization. Based on the equations above, it exists a relation between $P(k)$ and $\hat{P}(k)$ that

$$\begin{aligned} A_{d0} &= e^{\sum\limits_{i=1}^{3} P_i(k)A_iT_s} = T^{-1}\sum_{i=1}^{3} \hat{P}_i(k) e^{TA_iT^{-1}T_s} T \\ B_{d0} &= (\sum_{i=1}^{3} P_i(k)A_i)^{-1}(e^{\sum_{i=1}^{3} P_i(k)A_iT_s} - I)(\sum_{i=1}^{3} P_i(k)B_i) \\ &= \sum_{i=1}^{3} \hat{P}_i(k)A_i^{-1}T^{-1}(e^{TA_iT^{-1}T_s} - I)TB \end{aligned}$$

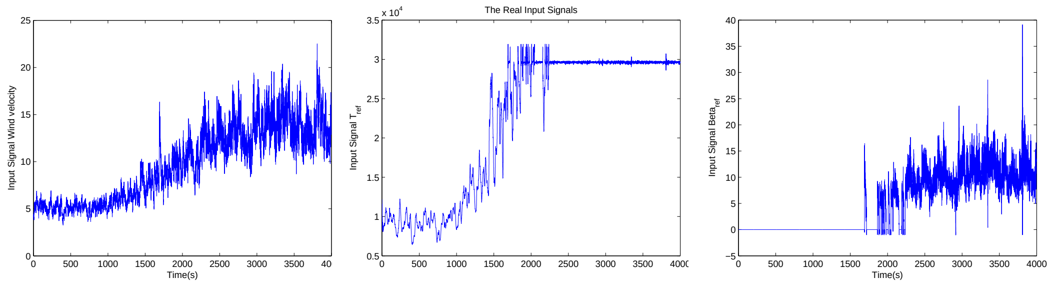


Figure 3.8: Input signals of the real system

When the estimated parameters $\hat{P}(k)$ are calculated, the terms A_{d0} and B_{d0} with respect to the real parameters $P(k)$ can be obtained. $P(k)$ can be recovered using reverse ZOH transformation given A_{d0} and B_{d0}. Therefore the real parameters of the original system are recovered from the estimated parameters.

3.5 Result Analysis

All of the data comes from the simulation benchmark, which simulates the real wind turbine system. The simulation time is 4000s and the sampling time is 0.01s, which results in a total data size of 400000 samples. The horizon window size is set to 600, which means in every $600 \times 0.01 = 6$s the parameter changes once. Due to the huge data size, the choice of horizon window size is a trade-off between the calculation cost time and result's accuracy. The input signals wind velocity $v_r(t)$, the reference signal for generator torque $T_{g,ref}$ and the reference signal for pitch angle β_{ref} are plotted versus time in Figure 3.8. It is noted that, in Figure 3.8, all input signals are not smooth. This is due to the noise in reality.

Using the wind turbine control strategy in Figure 3.2, the estimated parameters in the nonlinear system depend on the ranges of the wind velocity $v_r(t)$, the rotor speed $\omega_r(t)$ and the pitch angle $\beta(t)$ which are shown in Figure 3.9. The analysis on the model selection in Section 3.3 indicates that

- Model 1 $v_r \in [3.5, 8.5]$, $\beta = 0$, $\omega_r \in [0.75, 1.6]$;

- Model 2 $v_r \in [8.5, 11]$, $\beta = 0$, $\omega_r = 1.65$;

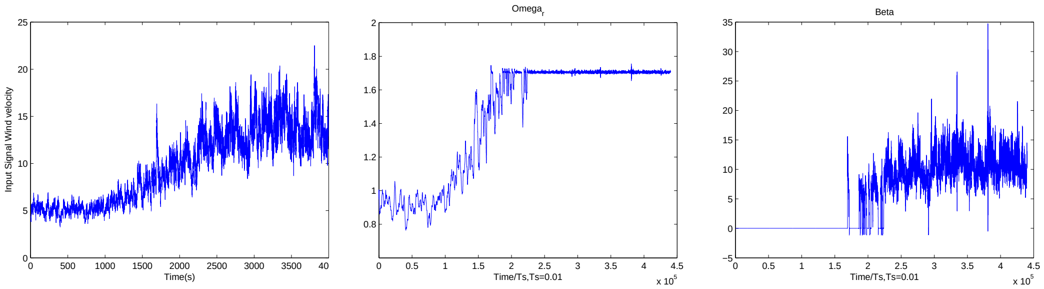


Figure 3.9: Determined variables of the control strategy in real system

- Model 3 $v_r \in [11, \infty)$, $\beta \neq 0$, $\omega_r \in [1.65, 1.8]$.

Compared with the data in Figure 3.9, the above ranges correspond to each model. Model 1 can be recognized approximately starting from 0s to 1750s, and from 1750s to 2000s the nonlinear model is approaching model 2. In the time range between 2000s and 4000s the nonlinear model is close to model 3.

The scheme works if the estimated parameters show the trend of the above analysis. For example, $P(k)$ is estimated to about $\begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$ when k starts from 0s to $1750 \div 6 \approx 292$s, $P(k)$ changes to a vector closed to $\begin{bmatrix}0 & 1 & 0\end{bmatrix}^T$ when $k \in [292, 333]$, and when k ranges from 333 to 666, $P(k)$ is closed to $\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$. However the estimated parameter $P(k) = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T, k \in [0, 666]$ which is not expected. This issue is probably caused by many factors, for example:

- The local models are so similar that the algorithm can not distinguish between them.
- The data processing in the similarity transformation and discretization can introduce some numerical errors.
- The data coming from the real benchmark includes overwhelming noise, which affects the accuracy of the algorithm.

From the state estimation point of view, the states generator torque T_g, pitch angle rate $\dot{\beta}$ and torsion angle of the drive train θ_Δ can not be measured directly from the benchmark. The errors between the estimated and the measured pitch angle β, generator speed ω_g, rotor speed ω_r are shown in Figure 3.10, Figure 3.11, and Figure 3.12.

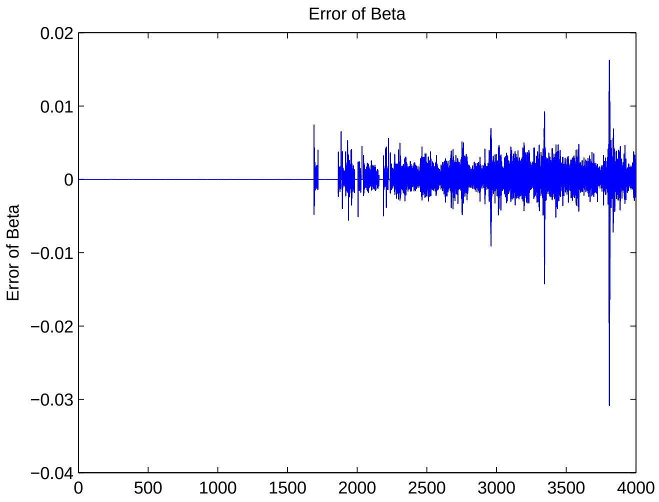


Figure 3.10: Error of state β estimation

The error range of the estimated angle pitch β follows the trend of the reference signal for β, staying within 0.03. When the range size of angle pitch is 30, the estimated angle pitch result is quite satisfactory.

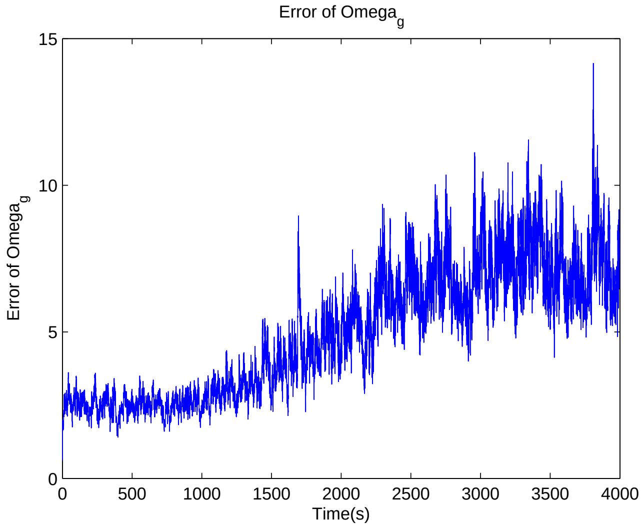


Figure 3.11: Error of state ω_g estimation

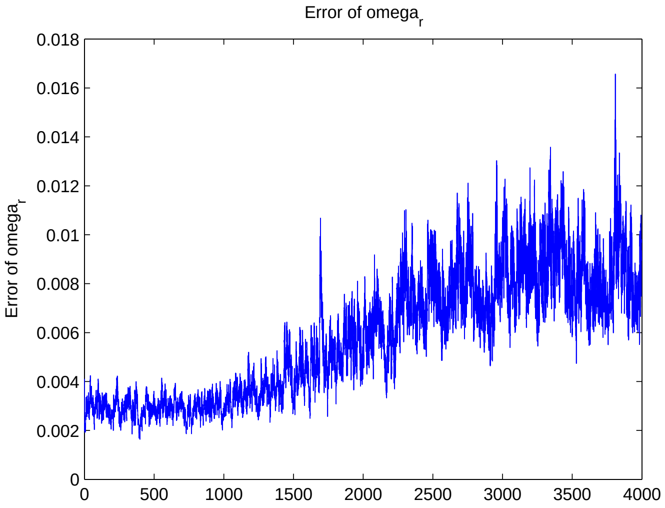


Figure 3.12: Error of state ω_r estimation

The range of the real generator speed ω_g is between 70 and 170 while the estimation error of ω_g remains under 15 which is relatively small. The range of the real rotor speed ω_r is from 0.8 to 1.8, and the corresponding estimation error with range 0.002 0.016 is tiny. The reason why the estimation errors of ω_g and ω_r follow the trend of wind velocity is the difference between the real and estimated models.

Chapter 4

Robust Fault Detection and Adaptive Threshold Evaluation

4.1 Introduction

The fault detection and isolation has attracted attention from industry and research fields for several decades as it can directly improve the reliability, efficiency, and safety of modern control systems such as trains, chemical plants, aircraft systems, and power plants. In generally, fault diagnosis methods can be classified into model-based and data driven types [39], in which the model-based type requires the model knowledge while the data driven type only calls for the process data. For each diagnosis method, there are four perspectives: quantitative and qualitative data-driven FD methods, as well as quantitative and qualitative model-based FD methods. Expert system and qualitative trend analysis are examples of data driven qualitative fault detection, while neural networks and principle component analysis belong to the quantitative data driven type [39] [40]. The digraphs method, fault tree method and qualitative physics method belong to the model-based qualitative classification. However, the most popular branch is the quantitative model-based methods which uses observer-based, parity space, and frequency domain methods [39] [40]. With the input and output signals and the relevant dynamic models, an indication which can distinguish between the nominal and faulty situations would be obtained by using the model-based methods

Within the quantitative model-based methods, the parity space approach to fault detection (FD) is a useful tool, especially for linear discrete time systems, and has been studied extensively [40–43]. The advantages of this approach over the observer/detection filter based FD approaches are simplicity and easy implementation for online applications [44]. A successful fault detection using the parity space approach relies heavily on the modeling accuracy, since the parity equations are obtained from analyzing the input/output relationship of the system [45]. Therefore, in the cases that has modeling errors, the design of parity space based FD becomes very challenging due to the accuracy issues related to the residual generation. As a result, the number of false alarms would increase dramatically when adopting the design without the proper treatment of the modeling errors. Compared with the robust FD design which is based on system state space models, fewer studies have been carried on parity space based methods which explicitly handles modeling errors [46, 47]. In [48], in addition to residual generation and evaluation, extensive work on fault signal estimation has also been conducted. In this thesis, the robust fault detection problem is addressed by integrating the design of a residual generator and the design of a threshold into an optimization based framework. The proposed approach explicitly handles the parabolic model uncertainties. Based on a similar idea as that in [48] [49], the major work of the chapter is focused on estimating the uncertainties in the normal system operation. The bounded threshold is thereby determined for fault detection and evaluation. The proposed scheme consists of three steps: the estimation of the modeling uncertainties, the determination of residual generator gains and the estimation of upper and lower bounds of the threshold. These three tasks are well transformed into three optimization problems with one LMI condition. The residual generator gain and the bounds of the threshold can be updated by using the sliding window technique. And the rate of false alarm can be dramatically reduced (even to zero) by applying this bounded adaptive threshold into the residual evaluation step.

The chapter illustrates the three sections: the modeling, the threshold generation and the simulations.

4.2 Modeling and Formulation

It is known that a discrete linear system can be described as a state space model:

$$\begin{aligned} x_{k+1} &= Ax_k + B_u u_k \\ y_k &= Cx_k + D_u u_k \end{aligned}$$

where $x_k \in \mathbb{R}^n$, $y_k \in \mathbb{R}^{n_y}$, $u_k \in \mathbb{R}^{n_u}$, $A \in \mathbb{R}^{n\times n}$, $B_u \in \mathbb{R}^{n\times n_u}$, $C \in \mathbb{R}^{n_y\times n}$ and $D_u \in \mathbb{R}^{n_y\times n_u}$. When $k = N$, the state variable and output can be expressed as

$$\begin{aligned} x_N &= A^N x_0 + A^{N-1}Bu_0 + \cdots + ABu_{N-2} + Bu_{N-1} \\ y_N &= CA^N x_0 + CA^{N-1}Bu_0 + \cdots + CABu_{N-2} + CBu_{N-1} + Du_N. \end{aligned}$$

The parity space model is then formed as

$$\begin{aligned} X_N &= C_{x0}x_0 + B_{xu}U_N \\ Y_N &= C_{y0}x_0 + D_{yu}U_N \end{aligned} \qquad (4.1)$$

where $X_N{=}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$, $Y_N{=}\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}$, $U_N{=}\begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$, $M_c{=}\begin{bmatrix} I \\ A \\ \vdots \\ A^{N-1} \end{bmatrix}$,
$C_{x0}{=}M_c \otimes A$, $C_{y0}{=}\mathrm{diag}\{C, \cdots, C\}M_c$,

$$B_{xu} = \begin{bmatrix} B_u & 0 & \cdots & 0 \\ AB_u & B_u & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}B_u & \cdots & AB_u & B_u \end{bmatrix}, D_{yu} = \begin{bmatrix} D_u & 0 & \cdots & 0 \\ CB_u & D_u & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{N-2}B_u & \cdots & CB_u & D_u \end{bmatrix}.$$

All of the calculations are carried out in a detection window, with a size of N.

The parity space model with additive fault can be built from

$$\begin{aligned} x_{k+1} &= Ax_k + B_u u_k + B_f f_k \\ y_k &= Cx_k + D_u u_k + D_f f_k \end{aligned}$$

to

$$\begin{aligned} X_N &= C_{x0}x_0 + B_{xu}U_N + B_{xf}F \\ Y_N &= C_{y0}x_0 + D_{yu}U_N + D_{yf}F \end{aligned}$$

where $f_k \in \mathbb{R}^{n_f}$, $F \in \mathbb{R}^{N \cdot n_f}$ are unknown fault signals and the corresponding $B_f \in \mathbb{R}^{n \times n_f}$, $D_f \in \mathbb{R}^{n_y \times n_f}$, $B_{xf} \in \mathbb{R}^{(N \cdot n) \times (N \cdot n_f)}$, $D_{xf} \in \mathbb{R}^{(N \cdot n_y) \times (N \cdot n_f)}$ are known matrices. Similar to the signal U_N and Y_N, F is constructed by $F = \begin{bmatrix} f_0 & f_1 & \cdots & f_{N-1} \end{bmatrix}^T$ and similar to B_{xu} and D_{yu}, $B_{xf} = \begin{bmatrix} B_f & 0 & \cdots & 0 \\ AB_f & B_f & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}B_f & \cdots & AB_f & B_f \end{bmatrix}$, $D_{yf} = \begin{bmatrix} D_f & 0 & \cdots & 0 \\ CB_f & D_f & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{N-2}B_f & \cdots & CB_f & D_f \end{bmatrix}$.

There is an assumption that in this detection window, all the uncertainties in the polytopic form (Equation 4.2) can be represented by the parameters a_i, $i = 1, ..., p$,

$$\begin{aligned} X_N &= (C_{x0} + \Delta C_{x0})x_0 + (B_{xu} + \Delta B_{xu})U_N + B_{xf}F \\ Y_N &= (C_{y0} + \Delta C_{y0})x_0 + (D_{yu} + \Delta D_{yu})U_N + D_{yf}F \end{aligned} \tag{4.2}$$

where

$$\begin{array}{lll} \Delta C_{y0} = \sum\limits_{i=1}^{p} a_i C_{y0}^i, & \Delta D_{yu} = \sum\limits_{i=1}^{p} a_i D_{yu}^i, & \Delta C_{x0} = \sum\limits_{i=1}^{p} a_i C_{x0}^i, \\ \Delta B_{x0} = \sum\limits_{i=1}^{p} a_i B_{xu}^i, & \sum\limits_{i=1}^{p} a_i = 1, & 0 \leq a_i \leq 1. \end{array}$$

The bounds on a_i can be acquired based on the model (4.2) in the condition of known $C_{x0}^i, C_{y0}^i, D_{yu}^i, B_{xu}^i$. Therefore, in a detection window with size of N, the range of the threshold, which is dependent on uncertainties, can then be estimated. The threshold dependent on a_i is adaptive, since the set of $a_i, \quad i = 1, ..., p$ would vary as the detection window moves forward.

4.3 Threshold Generation and Fault Detection

4.3.1 Iterative State and Parameter Estimation

A FD scheme essentially consists of two stages: threshold generation and fault detection. The fault information is not required in the first stage since the threshold only reflects the uncertainty and disturbance of a system. In this way, a two-level threshold can be designed according to the range of uncertainty. Moreover, in order to decrease the missed alarm rate and false alarm rate, one method is to maximize the lower bound on a_i (denoted as $\underline{a_i}$) and to minimize the upper bound on a_i (denoted as $\overline{a_i}$) according to the obtained Y, U and the bounds on the initial variable x_0. This method can be realized by solving two optimization problems as shown in Equation (4.3) and (4.4).

$$\begin{gathered}
\max_{a,x_0} \quad \underline{a_i}, \quad i = 1, \cdots, p \\
Y_N{=}C_{y0}x_0 + D_{yu}U_N + \sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0 \\
\underline{x_0} \le x_0 \le \overline{x_0} \\
\underline{a_i} \le e_i^T a \\
\sum_{i=1}^{p} a_i = 1 \\
0 \le a_i \le 1 \\
Y_N \textit{ and } U_N \textit{ are known}
\end{gathered} \tag{4.3}$$

$$\begin{gathered}
\min_{a,x_0} \quad \overline{a_i}, \quad i = 1, \cdots, p \\
Y_N{=}C_{y0}x_0 + D_{yu}U_N + \sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0 \\
\underline{x_0} \le x_0 \le \overline{x_0} \\
e_i^T a \le \overline{a_i} \\
\sum_{i=1}^{p} a_i = 1 \\
0 \le a_i \le 1 \\
Y_N \textit{ and } U_N \textit{ are known}
\end{gathered} \tag{4.4}$$

where $a{=}\begin{bmatrix} a_1 & a_2 & \cdots & a_p \end{bmatrix}^T$. The adaptive threshold is dependent on the varying upper and lower bounds on a_i when the detection window moves.

In the online FD implementation, the bounds on the state variable x_1 need to be estimated before the window shifting starts. Similarly, the bounds can be acquired by solving the following optimization problems (4.5) and (4.6),

$$\begin{gathered}
\max_{a,x_0} \quad \underline{x_{1,i}}, \quad i = 1, \cdots, n \\
X_N{=}C_{x0}x_0 + B_{xu}U_N + \sum_{i=1}^{p} a_i B_{xu}^i U_N + \sum_{i=1}^{p} a_i C_{x0}^i x_0 \\
\underline{x_{1,i}} \le e_i^T X_N, \quad i = 1, ..., n \\
\underline{x_0} \le x_0 \le \overline{x_0} \\
\underline{a} \le a \le \overline{a} \\
Y_N \textit{ and } U_N \textit{ are known}
\end{gathered} \tag{4.5}$$

$$\begin{gathered}
\min_{a,x_0} \quad \overline{x_{1,i}}, \quad i=1,\cdots,n \\
X_N{=}C_{x0}x_0 + B_{xu}U_N + \sum_{i=1}^{p} a_i B_{xu}^i U_N + \sum_{i=1}^{p} a_i C_{x0}^i x_0 \\
e_i^T X_N \le \overline{x_{1,i}}, \quad i=1,...,n \\
\underline{x_0} \le x_0 \le \overline{x_0} \\
\underline{a} \le a \le \overline{a} \\
Y_N \; and \; U_N \; are \; known
\end{gathered} \tag{4.6}$$

where the constraints that $\underline{a} \le a \le \overline{a}$ in (4.5) and (4.6) are estimated from optimization problems (4.3) and (4.4).

When the detection window, with a length size of N, shifts one sample forward, the input signal $U_N{=}\begin{bmatrix} u_0 & u_1 & \cdots & u_{N-1} \end{bmatrix}^T$ is updated with the new information $U_{N+1}{=}\begin{bmatrix} u_1 & u_2 & \cdots & u_N \end{bmatrix}^T$ and the corresponding output signal $Y_N{=}\begin{bmatrix} y_0 & y_1 & \cdots & y_{N-1} \end{bmatrix}^T$ is updated with $Y_{N+1}{=}\begin{bmatrix} y_0 & y_1 & \cdots & y_{N-1} \end{bmatrix}^T$. Using the updated input/output signals (i.e. U_{N+1}, Y_{N+1}) and the estimated bounds on the initial state variable value (i.e. x_1), the bounds on the new coefficient a in a new detection window can be obtained by optimization with (4.3) and (4.4) in an online iterative process.

The equations, $Y_N{=}C_{y0}x_0 + D_{yu}U_N + \sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0$ in (4.3) and (4.4) and $X_N{=}C_{x0}x_0 + B_{xu}U_N + \sum_{i=1}^{p} a_i B_{xu}^i U_N + \sum_{i=1}^{p} a_i C_{x0}^i x_0$ in (4.5) and (4.6), with respect to x_0 and a, are nonlinear, non-convex and non-concave, which means the optimization problems (4.3)-(4.6) can not be solved by regular optimization tools and methods.

Only acquiring the maximization of the lower bound is illustrated here, since the minimization of the upper bound is obtained in the similar way. Define $\mathbb{R}_+^m := \{v \in \mathbb{R}^m : v \ge 0\}$ and denote e_i as the i^{th} column vector of an identity matrix $I_{p\times p}$. Here the parameters, $D_x \in \mathbb{R}_+^{n\times n}$, $\mu_y \in \mathbb{R}^{N\cdot n_y}$, $D_a \in \mathbb{R}_+^{p\times p}$ and $D_b \in \mathbb{R}$, are introduced.

Since $\underline{a_i}$ is the lower bound on a_i, $\underline{a_i} - e_i^T a \le 0$ always exists. There is an equation transformation that

$$\underline{a_i} - e_i^T a = -(x_0 - \underline{x_0}) D_x (\overline{x_0} - x_0) - \mu_y^T (Y - C_{y0} x_0 - D_{yu} U - \sum_{i=1}^{p} a_i D_{yu}^i U - \sum_{i=1}^{p} a_i C_{y0}^i x_0) - a^T D_a (e - a) - D_b (1 - e^T a)^T - \begin{bmatrix} x_0^T & a^T & 1 \end{bmatrix} \Gamma \begin{bmatrix} x_0 \\ a \\ 1 \end{bmatrix} \leq 0$$

where the symmetric matrix

$$\begin{aligned} \Gamma &= \begin{bmatrix} D_x & \frac{1}{2} C_{y0m}^T \mu_k^T & \Xi_1^T \\ \frac{1}{2} \mu_k C_{y0m} & D_a & \Xi_2^T \\ \Xi_1 & \Xi_2 & \Xi_3 \end{bmatrix}, \\ \Xi_1 &= -\tfrac{1}{2} \underline{x_0^T} D_x + \tfrac{1}{2} \mu_y^T C_{y0} - \tfrac{1}{2} \overline{x_0}^T D_x^T, \\ \Xi_2 &= \tfrac{1}{2} D_b e^T - \tfrac{1}{2} e_i^T + \tfrac{1}{2} u^T D_{yum}^T \mu_k^T - \tfrac{1}{2} e^T D_a^T, \\ \Xi_3 &= \underline{x_0^T} D_x \overline{x_0} - \mu_y^T y - D_b + \mu_y^T D_{yu} u + \underline{a_i} \end{aligned}$$

In matrix Γ,

$$e^T = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^p, \qquad \mu_k = \begin{bmatrix} \mu_y^T & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mu_y^T \end{bmatrix}_{p \times (p \cdot n_y \cdot N)},$$

$$C_{y0m} = \begin{bmatrix} C_{y0}^1 \\ \vdots \\ C_{y0}^p \end{bmatrix}_{(p \cdot n_y \cdot N) \times n}, \quad D_{yum} = \begin{bmatrix} D_{yu}^1 \\ \vdots \\ D_{yu}^p \end{bmatrix}_{(p \cdot n_y \cdot N) \times (N \cdot n_u)}.$$

It follows that a sufficient condition for the optimization problem (4.3) is

$$\begin{gathered} \max \quad \underline{a_i} \\ \Gamma(\underline{a_i}, D_x, D_a, D_b, \mu_y) \succcurlyeq 0 \\ D_x \in \mathbb{R}_+^{n \times n}, D_a \in \mathbb{R}_+^{p \times p} \\ \mu_y \in \mathbb{R}^{N \cdot n_y}, D_b \in \mathbb{R}. \end{gathered} \tag{4.7}$$

The detailed proof of sufficiency for (4.3) is presented in [50]. The optimization problem (4.7) also follows Farkas' Lemma [51]. The optimization problem (4.7) could also be solved by using the LMI technique instead of solving (4.3).

The optimization problems (4.4), (4.5) and (4.6) can be treated as three LMI problems in a similar manner. As long as the best estimated a is calculated, the tightest threshold would be designed to decrease the miss alarm rate and false alarm rate. The estimated lower and upper bounds on x_1 would be considered as the bounds for the initial state x_0 in the detection window of next step estimation.

4.3.2 Threshold Selection and Fault Detection

The idea behind the model-based FD is to design a threshold using the redundancy in the information of measurement and process model [52] [53]. If the designed indication does not match the expected range (also called threshold), a fault is detected.

The two desired properties in any FD method are: robustness and sensitivity [54]. Robustness means the FD algorithm does not produce false alarms due to disturbances and uncertainties, while sensitivity means the FD algorithm does not generate missed alarms even in the condition of a small fault. Threshold generation should satisfy the above properties, and decrease both the false alarm rate and missed alarm rate at the same time, which requires the threshold to be bounded as tightly and accurately as possible.

Two-Level Threshold Design

In the threshold selection stage, we aim to estimate the maximum and minimum residual signals of the unknown model uncertainties in the fault-free case. At every time instant, except for the first one, the range of x_0 is re-calculated, and it may be influenced by uncertainty and faults in the system. In order to avoid this, the residual signal is designed as:

$$r_N = \|K_N(Y_N - D_{yu}U_N)\|_2$$

where $K_N = K_{N1}K_{N2}$. $K_{N2} \in \mathbb{R}^{n\times(N\cdot n_y)}$ is selected from a set of basis for null space of C_{y0}. If and only if $Nn_y \geq n$, a non-zero matrix K_{N2} exists due to $C_{y0} \in \mathbb{R}^{(n_y\cdot N)\times n}$. A proper $K_{N1} \in \mathbb{R}^{n\times n}$ can be selected by

$$\sup_{K_{N1}} \frac{K_{N1}K_{N2}D_{yf}}{K_{N1}K_{N2}(\sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0)}.$$

To avoid decoupling the fault term, K_N also satisfies $K_N D_{yf} \neq 0$.

To sum up, a proper K in this approach should have the property of

$$\begin{array}{l} \sup\limits_{K_{N1}} \frac{K_{N1}K_{N2}D_{yf}}{K_{N1}K_{N2}(\sum\limits_{i=1}^{p} a_i D_{yu}^i U_N + \sum\limits_{i=1}^{p} a_i C_{y0}^i x_0)} \\ K_{2N} \cdot C_{y0} = 0 \\ K_N \cdot D_{yf} \neq 0. \end{array}$$

At every time instant, the scalar indication in the fault-free case, corresponding to a detection window, is the so called threshold . Therefore the threshold is designed as

$$\begin{aligned} J_N &= \|K_N(Y_N - D_{yu}U_N)\|_2 \\ &= \|K_N(C_{y0}x_0 + \sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0)\|_2 \\ &= \|K_N(\sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0)\|_2. \end{aligned}$$

The two-level threshold is obtained by maximization and minimization of J_N subject to the ranges of a and x_0.

Fault Detection

The criterion of fault detection is

$$\begin{cases} \|r_N\|_2 \notin (\underline{J}, \overline{J}), & \text{faulty case} \\ \|r_N\|_2 \in (\underline{J}, \overline{J}), & \text{fault free case.} \end{cases}$$

The residual with fault is calculated by

$$\begin{aligned} \|r_{Nf}\|_2 &= \|K_N(Y_N - D_{yu}U_N)\|_2 \\ &= \|K_N(C_{y0}x_0 + \sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0 + D_{yf}F)\|_2 \\ &= \|K_N(\sum_{i=1}^{p} a_i D_{yu}^i U_N + \sum_{i=1}^{p} a_i C_{y0}^i x_0 + D_{yf}F)\|_2 \end{aligned}$$

If the calculated $\|r_{Nf}\|_2 \notin (\underline{J}, \overline{J})$, then the fault is detected.

Compared with the traditional single-level threshold, the two-level threshold in this approach is able to decrease the false alarm rate and missing alarm rate effectively, provided that the single-level threshold is set in the middle of the two-level threshold. A false alarm appears when the indication jumps through threshold which is caused by an unexpected uncertainty, but the two-level threshold scheme considers it as fault-free. a missed alarm occurs, when the indication is inside of the one-level threshold but is smaller than lower bound of the two-level threshold. However, this fault can be detected out in the two-level threshold scheme.

4.4 Simulation Results

The generation of the threshold and fault detection scheme is summarized in the flow chart of Figure 4.1:

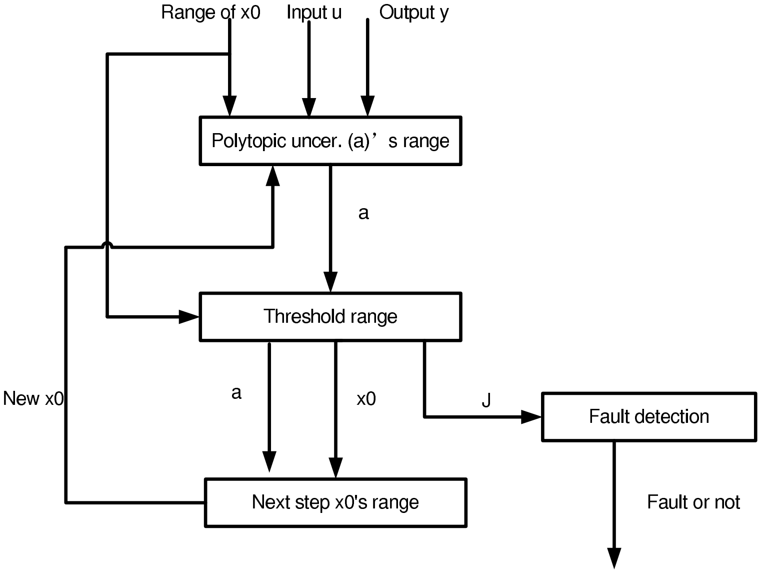


Figure 4.1: Flow chart depicting the generation of threshold and fault detection

To illustrate this approach, a numerical example is simulated. In order to prove this approach does not work for one particular system, but works for any proper one, the parameters (i.e. A, B_u, C, D_u, B_f, D_f etc.) and signals (i.e. $u_i, y_i, \overline{x_0}, \underline{x_0}$ etc.) are generated randomly with compatible dimension. The detection window size $N = 4$, state variable dimension $n = 2$; input dimension $n_u = 2$; output dimension $n_y = 4$, fault dimension $n_f = 2$, shifting number is 8 and the polytopic uncertainty number is $p = 3$. In this example, the evaluation of coefficients of uncertainty $a_1, a_2, a_3 (p = 3)$ is carried out by solving optimization (3.3) and (3.4). The result is shown in Table 4.1.

In Table 4.1, the estimated upper and lower bounds on a_i (i.e. a_1, a_2, a_3) are

Table 4.1: EVALUATION OF POLYTOPIC UNCERTAINTIES

	Coefficient of Uncer	Upper bound	True value	Lower bound
$k_{shift} = 1$	a_1	0.1182	0.1000	0.0799
	a_2	0.5087	0.5000	0.4929
	a_3	0.4083	0.4000	0.3994
$k_{shift} = 2$	a_1	0.1999	0.2000	0.2001
	a_2	0.6000	0.6000	0.6000
	a_3	0.2000	0.2000	0.2000
$k_{shift} = 3$	a_1	0.2999	0.3000	0.3002
	a_2	0.2000	0.2000	0.2001
	a_3	0.5000	0.5000	0.5001
$k_{shift} = 4$	a_1	0.0998	0.1000	0.1001
	a_2	0.0998	0.1000	0.1002
	a_3	0.7998	0.8000	0.8002

very close and tight. It means a satisfactory polytopic uncertainty evaluation has been obtained.

When the detection window moves, x_1, which is also x_0 in the next step, is estimated by solving the optimization problem (4.5) and (4.6), which are subject to the range of the last step's x_0 and uncertainty a_1, a_2, a_3. In this example, the initial variable is expressed as $x_0 = \begin{bmatrix} x_0^1 & x_0^2 \end{bmatrix}^T$ with a length of 2. The evaluations of x_0^1 and x_0^2 are listed in the first three and last three columns of Table 4.2. As one time instant goes by, a new initial state variable needs to be calculated to generate a threshold. Table 4.2 is listed in the shifting order.

Table 4.2: EVALUATION OF INITIAL STATE VARIABLE

	The first state of x_0			The second state of x_0		
Shift steps	Lower bound	True value	Upper bound	Lower bound	True value	Upper bound
1	6.783	7.822	9.729	0.705	2.662	3.862
2	-11.919	-10.58	-10.379	-0.361	0.998	2.681
3	30.654	31.449	33.636	-27.302	-25.527	-23.949
4	-45.727	-42.680	-41.783	45.312	48.976	51.863
5	53.581	55.896	61.030	-44.690	-40.252	-37.137
6	-45.446	-39.445	-35.831	25.135	33.201	38.523
7	11.630	15.745	20.707	-11.011	-6.189	1.546
8	-4.519	-2.220	0.995	-33.987	-28.602	-25.726

At every shift instant there exists an indication $r_N = K_N(Y_N - D_{yu}U_N)$, because the system is discrete-time. Figure 4.2 illustrates there is no fault in the system from shift 1 to 8.

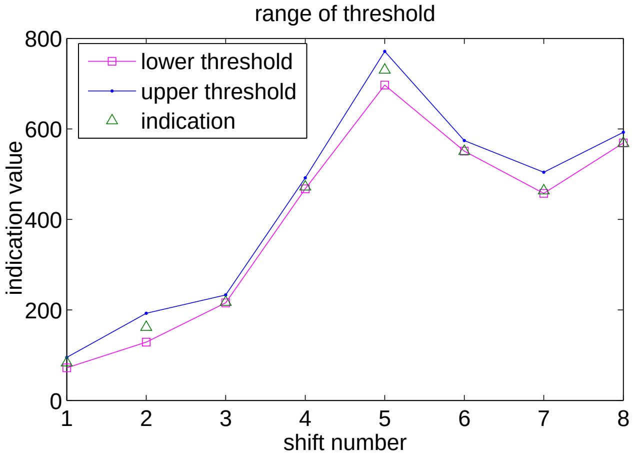


Figure 4.2: Adaptive threshold and fault detection without fault

In the example $n_f = 2$ and $N = 4$, so the fault detection has the form $F = \begin{bmatrix} f_1 & f_2 & f_3 & f_4 \end{bmatrix}^T = \begin{bmatrix} f_1^1 & f_1^2 & \cdots & f_4^1 & f_4^2 \end{bmatrix}^T$, where $f_i = \begin{bmatrix} f_i^1 & f_i^2 \end{bmatrix}$. The subscript i is the shift time and the superscript to isolate the faults. There are eight elements in the fault vector F at one shift time. As long as one element is nonzero, the indication at that shift number is outside of $\begin{bmatrix} \underline{J} & \overline{J} \end{bmatrix}$, which means faults exists in the detection window.

The following three cases are simulated to verify the advantages of this approach.

Case 1

$f_k^1 = 0$ for all k, $f_k^2 = \begin{cases} 1, & k = 1 \\ 0, & k \neq 1 \end{cases}$

In Figure 4.3, it shows that f_k^1 and f_k^2 respectively have four elements in the detection window. At shift one, all the elements of $f_k^1 = 0$ are zero and f_k^2 includes one non-zero element at the first place of the detection window. At shift two, f_k^1 still contains zero elements, however the non-zero element of f_k^2 at the first place is moved out of the detection window, and then another zero at the last place makes a supplement.

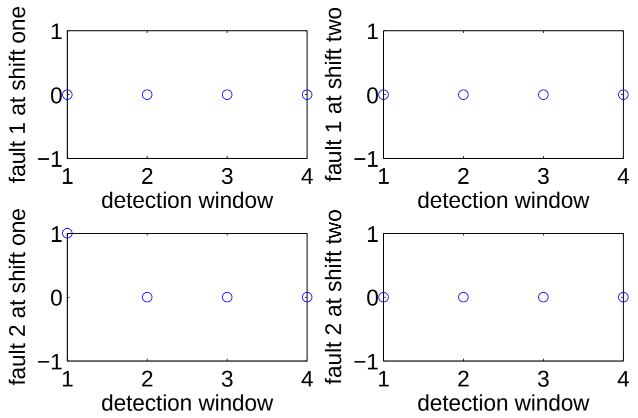


Figure 4.3: Fault 1 and 2 at first and second shifts in case 1

From the second to eighth shift time there is no fault in both f_k^1 and f_k^2, which have the same situation as the second shift, so all the indications in these shifts are inside the adaptive threshold bound except for the first shift, where a fault exists. The adaptive threshold and detection result of Case 1 is shown in Figure 4.4.

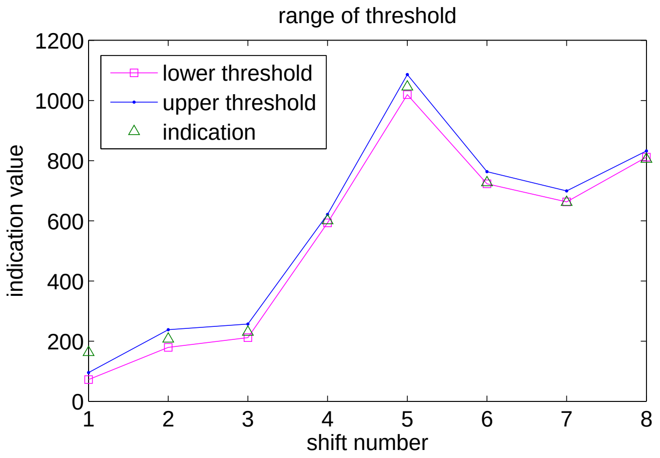


Figure 4.4: Fault detection in case 1

Case 2

$$f_k^1 = \begin{cases} 1, & k = 1, 2, 3 \\ 0, & otherwise \end{cases}, f_k^2 = \begin{cases} 1, & k = 1, 3 \\ 0, & otherwise \end{cases}$$

Figure 4.5 presents f_k^1 and f_k^2 at the first four shifts. The first elements $f_1^1 = 1$ and $f_1^2 = 1$ from first shift to second shift are moved out of the detection window, and

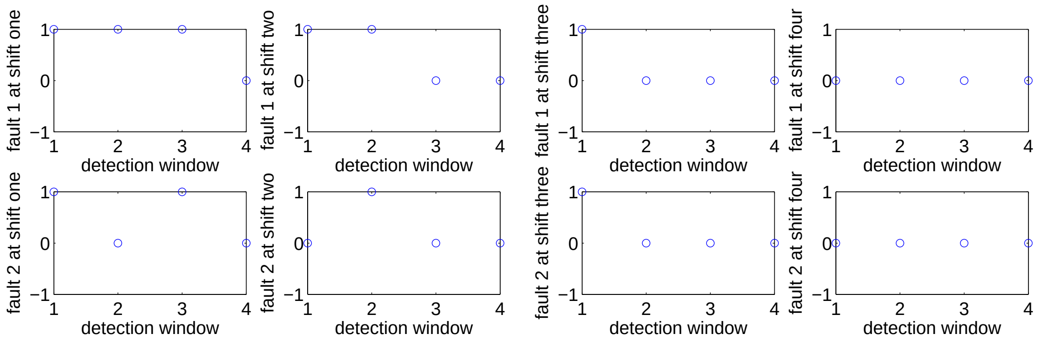


Figure 4.5: Fault 1 and fault 2 at the first 4 shifts in case 2

then followed by $f_5^1 = 0$ and $f_5^2 = 0$ at the last place.

In Figure 4.5, there faults exist from the first shift to the third shift and there is no fault until the fourth shift. Results from adaptive threshold, FD detection are illustrated in Figure 4.6.

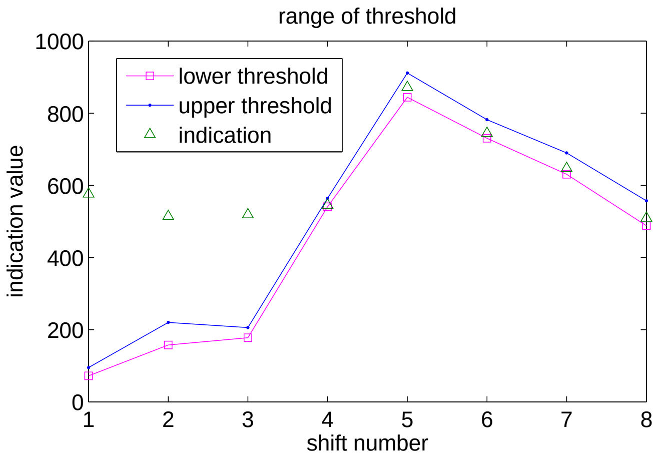


Figure 4.6: Fault detection in case 2

In Figure 4.6, the detection windows corresponding to shift one, two and three contain the faults. The indications are outside of the adaptive threshold bounds in these three shifts.

Case 3

$$f_k^1 = \begin{cases} 1, & k = 4 \\ 0, & otherwise \end{cases}, f_k^2 = \begin{cases} 1, & k = 3 \\ 0, & otherwise \end{cases}$$

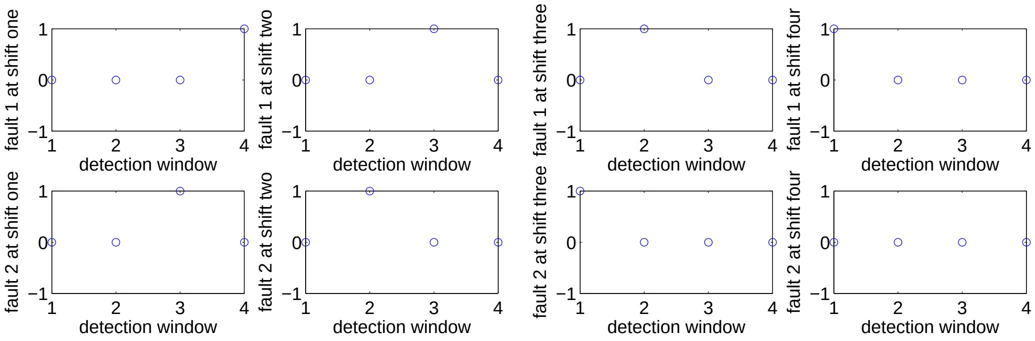


Figure 4.7: Fault 1 and fault 2 at the first 4 shifts in case 3

In Figure 4.7, there exits some faults from the first shift to the fourth shift. $f_4^1 = 1$ will move out of the detection at the fifth shift. FD using the adaptive threshold is illustrated in Figure 4.8, which demonstrates this approach can provide a satisfactory result.

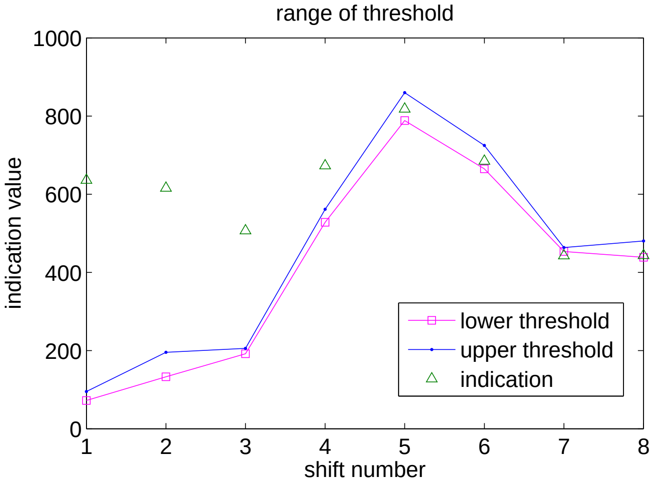


Figure 4.8: Fault detection in case 3 with faults in shift 1, 2, 3, 4

Chapter 5

Conclusion

The last chapter of the thesis provides a conclusion of the thesis and possible future improvements.

5.1 Summary of Thesis Work

The thesis describes a new design which can online estimate the states and parameters iteratively in an LPV system. The LPV model with polytopic system matrices composed of the varying estimated parameters and the local models can approximate a nonlinear model. The local models are obtained using Jacobian linearization around the operating points, the selection of which can influence the approximation effect to the nonlinear model. In general, the larger quantity of the selected local models, the better the approximation effect would be.

In this thesis, an online scheme is designed to estimate the parameters using optimization and the states through a state observer for an LPV system. These two estimations are carried out in different sliding windows. In the state sliding window, state estimation uses the Luenberger observer under a model with the parameters, which are obtained through optimization in the parameter window. Iteratively the parameters in the next estimation window are estimated based on the previously estimated state variables. The proposed state and parameter estimation scheme is realized in an industrial nonlinear rig system and verified by the accurate simulation results. Furthermore, the scheme is also applied to a complicated nonlinear industrial system wind turbine system. The wind turbine system consists of four subsystems: blade and pitch, drive train, generator and converter, and controller subsystem. The operating locus of a typical wind turbine has three subareas. As illustrated in Figure 3.2, in the first subarea, the pitch actuation does not take effect and the pitch angle stays at the maximum while the rotor speed increases as the wind speed increases. In the second subarea, both the pitch angle and the rotor speed is kept constant and are independent of the wind speed. In the last subarea, the pitch angle begins to grow in order to keep a reasonable rotor speed when facing an overrated wind speed. On the basis of the three subareas, the nonlinear wind turbine system can be considered as three linear models regarding the different wind speeds. Therefore, the wind turbine system is an ideal simulation object for the parameters and state estimation scheme for an LPV system. The key issue in this simulation is how to obtain the local models corresponding to the different control subareas. The nonlinearity of wind turbine system lies in a one-to-one table instead of a nonlinear analytic function, so the numerical linearization method around the operating points is adopted. Since the scheme is carried out under the discrete time model, reverse transformation of zero-order hold discretization can be use for recovering the original parameters of the continuous time model from the estimated parameters in the discrete time model.

The thesis also provides an online fault detection scheme based on parity space. The parity space model is formed by integrating several linear state space models with successive discrete time indexes as a matrix equation. In this scheme, a polytopic uncertainty and an addictive fault are considered in the model. The ranges of them are obtained by solving optimization problems based on the input/output measurements in two different estimation windows. The bounds on the uncertainty parameters are based on the estimated ranges of state variable, which are obtained in the same way based on the model with the estimated bounds on uncertainty parameters of the previous window. However, the nonlinear optimization problems can not be solved by the standard optimization technique. A solvable linear matrix inequality (LMI) problem is targeted instead of the original nonlinear optimization since the solutions of the LMI problem imply the ones of the original optimization problem. Based on the ranges of parameters and state variables in each loop, a two-level adaptive threshold is designed for online fault detection, and adaptively varies with the sliding window. The proposed faults detection scheme based on the parity space method has been verified by the simulation results.

5.2 Future Work

The thesis proposes a method of parameters and state estimation for an LPV system, an application for wind turbine system, and an online fault detection scheme based on the parity space model. Based on the current work, several possible directions for future research and design refinements are listed as follows:

- The parameters and state estimation for LPV systems scheme can be applied only in a low rate LPV system, such as process control system, since the parameters in the LPV model are approximated as constants in the estimation window.

- In this scheme, noise is not considered in the model, and the Luenberger state observer is adopted. Another direction could be to use a robust observer instead of the Luenberger observer, so as to reduce the influence of disturbance and noise in the models.

- In the wind turbine application, the tower influence is neglected in the benchmark. As a result, the target nonlinear state space model has less order than the actual nonlinear state space.

- In the application of wind turbine system, after substituting all the parameter values into the nonlinear state space model, the system matrix is ill-conditioned. Therefore, the similar local linear models still cannot be distinguished due to the similarity of the local models, even after balancing. The unexpected parameter estimation is partially resulted from it.

- The nonlinearity of the wind turbine model lies in the one-to-one lookup table instead of an analytic function. Usually the exact operating points are not listed in the table, thus the nearest numbers around the operating points are adopted in the numerical linearization process. That results in the biased linearized models, compared with using an analytic function.

According to the above work yet to be done, the work could be improved in future.

Bibliography

[1] H. Alwi, C. Edwards, and A. Marcos, “Fault reconstruction using an LPV sliding mode observer for a class of LPV systems,” *Journal of the Franklin Institute*, vol. 4, no. 2, pp. 168–180, 2011.

[2] A. Varga, A. Hansson, and G.Puyou, *Optimization Based Clearance of Flight Control Laws*, ser. Lecture Notes in Control and Information Sciences. Berlin, Heidelberg: Springer, 2012, vol. 416.

[3] D. A. Lawrence and W. J. Rugh, “Input-output pseudo linearization for nonlinear systems,” *IEEE Transactions on Automatic Control*, vol. 39, no. 11, pp. 451–462, 1994.

[4] D. G. Luenberger, “Observing the state of a linear system,” *IEEE Transactions on Military Electronics*, vol. 8, no. 2, pp. 74–80, 1964.

[5] J. Birk and M. Zeitz, “Extended luenberger observer for non-linear multivariable systems,” *International Journal of Control*, vol. 147, no. 6, pp. 1823–1835, 1988.

[6] R. E. Kalman, “A new approach to linear filtering and prediction problem,” *Journal of basic Engineering*, pp. 1823–1835, 1960.

[7] A. H. Jazwinski, *Stochastic process and filtering Theory*. New York: Academic Press, 1970.

[8] A. Zolghadri, D. Henry, and S. Grenaille, “Fault diagnosis for LPV systems,” in *16th Mediterranean Conference on Control and Automation*, Ajaccio, France, June 2008.

[9] S. Grenaille, D. Henry, and A. Zolghadri, “A method for designing FDI filters for polytopic LPV models,” in *SAFE PROCESS 2006*, Beijing, China, 2006.

[10] J. Bokor and G. Balas, “Detection filter design for LPV systems,” *Automatica*, vol. 40, pp. 511–518, 2004.

[11] R. Hallouzi, V. Verdultand, R. Babuska, and M. Verhaegen, “Fault detection and identification of actuator faults using linear parameter varying models,” in *Preprint 16th IFAC World congress*, Prague, Czech Republic, June 2005.

[12] J. Bokor, Z. Szabo, and G. Stikkel, “Failure detection for quasi LPV systems,” in *IEEE Conference on Decision and Control*, vol. 3, Las Vegas, Nevada, USA, 2002, pp. 3318–3323.

[13] S. Grenaille, D. Henry, and A. Zolghadri, “A method for designing fault diagnosis filters for LPV polytopic systems,” *Journal of Control Science and Engineering*, 2008.

[14] S. Armeni, A. Casavola, and E. Mosca, “Robust fault detection and isolation for LPV systems under a sensitivity constraint,” *International Journal of Adaptive Control and Signal Processing*, vol. 23, pp. 55–72, 2009.

[15] Y. Hu, “Identification and state estimation for linear parameter varying systems with application to battery management,” Ph.D. dissertation, Electrical and Computer Engineering, Ohio State University, 2010.

[16] X. G. Yan and C. Edwards, “Nonlinear robust fault reconstruction and estimation using a sliding mode observer,” *Automatica*, vol. 43, no. 9, pp. 1605–1614, 2007.

[17] F. Blanchini and S. Miani, “Stabilization of LPV systems: state feedback, state estimation and duality,” in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 2, Maui, Hawaii USA, December 2003, pp. 1492–1497.

[18] C. E. Garcia, D. M. Prett, and M. Morari, “Model predictive control: theory and practice-a survey,” *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[19] M. Morari and J. H. Lee, “Model predictive control: past, present and future,” *Computers and Chemical Engineering*, vol. 23, no. 3, pp. 667–682, 1999.

[20] E. Polak and T. H. Yang, “Moving horizon control of linear systems with input saturation and plant uncertainty,” *International Journal of Control*, vol. 53, no. 4, pp. 613–638, 1993.

[21] E. Zafiriou, “Robust modelpredictive control of processes with hard constraints,” *Computer and Chemical Engineering*, vol. 14, no. 2, pp. 359–371, 1990.

[22] L. Imsland, J. A.Rossiter, B. Pluymers, and J. Suykens, “Robust triple mode MPC,” *International Journal of Control*, vol. 81, no. 5, pp. 679–689, 2008.

[23] M. V. Kothare, V. Balakrishnan, and M. Morari, “Constrained model predictive control using linear matrix inequalities,” *International Journal of Control*, vol. 32, no. 3, pp. 1361–1379, 1996.

[24] B. Kouvaritakis, J. A. Rossiter, and J. Schuurmans, “Efficient robust predictive control,” *International Journal of Control*, vol. 45, no. 5, pp. 1545–1549, 2000.

[25] Y. Lee and B. Kouvaritakis, “Constrained robust model predictive control based on periodic invariance,” *Automatica*, vol. 42, no. 5, pp. 2175–2181, 2006.

[26] W. Mao, “Robust stabilization of uncertain time-varying discrete systems and comments on an improved approach for constrained robust model predictive control,” *Automatica*, vol. 39, no. 6, pp. 1109 – 1112, 2003.

[27] L. R. Hunt, R. Sastry, and G. Meyer, “Global transformations of nonlinear control systems,” *IEEE Transaction on Automatic Control*, vol. 37, no. 3, pp. 392–298, 1992.

[28] B. Jakubczyk and W. Respondek, *On linearization of control systems*. Bulletin Academic Polonaise des Sciences, 1980.

[29] A. Banerjee, Y. Arkun, B. Ogunnaike, and R. Pearson, “Estimation of nonlinear systems using linear multiple models,” *Process systems Engineering*, vol. 43, no. 5, pp. 192–198, 1997.

[30] G. J. Preston, D. N. Shields, and S. Daley, “Application of a robust nonlinear fault detection observer to a hydraulic system,” in *UKACC International Conference*, vol. 14, 1996, pp. 1484–1489.

[31] H. E. Merritt, *Hydraulic Control Systems*. John Wiley & Sons, 1967.

[32] J. Watton, *Modelling, monitoring and diagnostic techniques for fluid power systems*. Springer, 2007.

[33] C. Gao, Q. Zhao, and G. Duan, “Multiplicative fault estimation for a type of nonlinear uncertain system,” in *Control Conference (CCC), 2011 30th Chinese*, Yantai, China, July 2011, pp. 4355 –4360.

[34] P. Gevorkian, *Wind Energy Technologies in Alternative Energy Systems in Building Design*. McGraw-Hill Press, 2010.

[35] F. D. Bianchi, R. J. Mantz, and C. F. Christiansen, “Gain scheduling control of variable-speed wind energy conversion systems using quasi LPV models,” *Control Engineering Practice*, vol. 13, no. 15, pp. 247–255, 2005.

[36] T. Esbensen and C. Sloth, “Fault diagnosis and fault-tolerant control of wind turbines,” Master’s thesis, Aalborg University, 2009.

[37] J. Mohammadpour and C. W. Scherer, *Control of Linear Parameter Varying Systems with Applications*. Springer, 2012.

[38] P. F. Odgaard, "Fault tolerant control of wind turbine bench mark model," kk-electronic, Tech. Rep., August 2009.

[39] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, "A review of process fault detection and diagnosis part iii: Process history based methods," *Computers and Chemical Engineering*, vol. 27, no. 8, pp. 327–346, 2003.

[40] J. Gertler, *Fault Detection and Diagnosis in Engineering System*. CRC Press, 1998.

[41] X. Ding, L. Guo, and T. N. Jeinsch, "A characterization of parity space and its application to robust fault detection," *IEEE Transaction on Automatic Control*, vol. 44, no. 4, pp. 337–343, 1999.

[42] E. Y. Chow and A. S. Willsky, "Analytical redundancy and the design of robust failure detection system," *IEEE Transaction on Automatic Control*, vol. 27, pp. 603–614, 1984.

[43] A. Hagenblad, F. Gustafsson, and I. Klein, "A comparison of two method for stochastic fault detection: the parity space approach and principal component analysis," in *IFAC Symposium on System Identification*, vol. 16, Rotterdam, Netherlands, 2003, pp. 1284–1289.

[44] J. Chen, R. J. Patton, and H. Zhang, "Design of unknown input observers and robust fault detection filters," *International Journal of Control*, vol. 63, pp. 86–105, 1996.

[45] F. Gustafsson, "Statistical signal processing approaches to fault detection," in *6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, vol. 5, Beijing, China, 2006, pp. 584–592.

[46] Z. Li and I. M. Jaimoukha, "Observer-based fault detection and isolation filter design for linear time-invariant system," *International Journal of Control*, vol. 82, pp. 171–182, 2009.

[47] S. X. Ding, P. Zhang, and P. M. Frank, "Threshold calculation using lmi-technique and its integration in the design of fault detection system," in *42nd IEEE Conference on Decision and Control*, vol. 17, Maui, Hawaii, USA, December 2003, pp. 284–392.

[48] M. J. D. Powell, *A fast algorithm for nonlinearly constrained optimization calculation, Numerical Analysis*. Springer, 1978.

[49] R. J. Patton and J. Chen, "Observer-based fault detection and isolation: robustness and applications," *Control Engineering Practice*, vol. 5, pp. 671–682, 1997.

[50] Z. Zhang and I. M. Jaimoukha, "Fault detection and isolation for linear discrete-time systems using input/output measurement analysis," in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, China, December 2009, pp. 4908 – 4913.

[51] R. Jagannathan and S. Schaible, "Duality in generalized fractional programming via Farka's lemma," *Journal of optimization theory and applications*, vol. 41, no. 3, pp. 147–158, 1983.

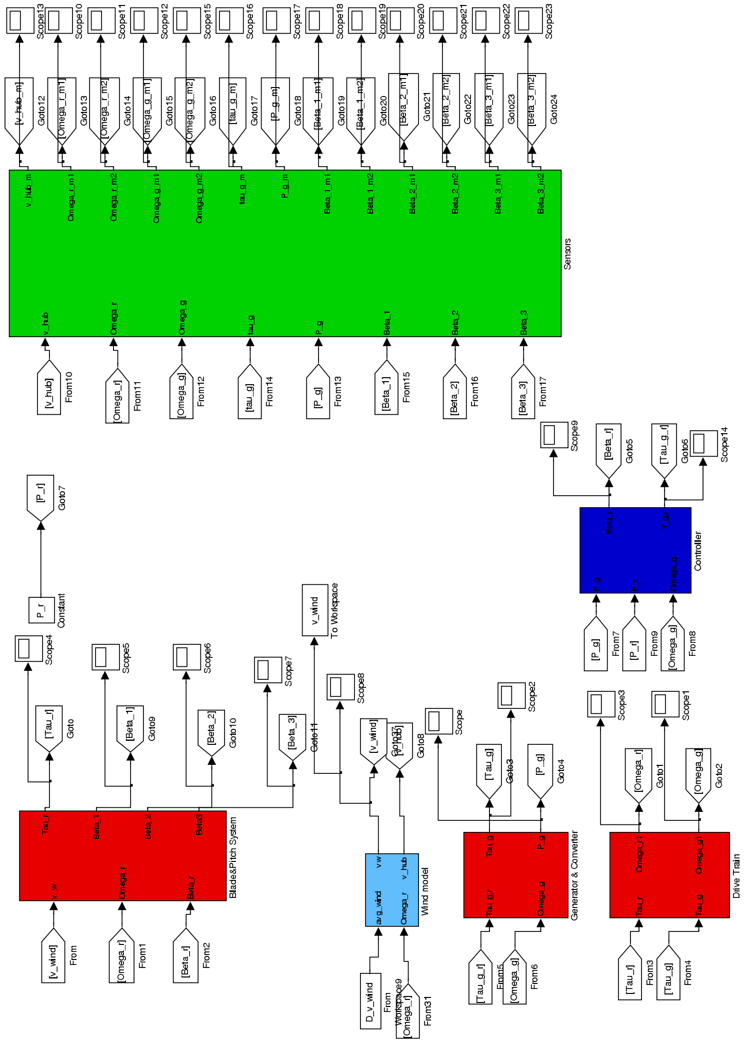
[52] F. Previdi and M. Lovera, "Identification of nonlinear parametrically varying models using separable least squares," *International Journal of Control*, vol. 77, no. 16, pp. 1382–1392, 2004.

[53] M. Nemani, R. Ravikanth, and B. Bamieh, "Identification of linear parametrically varying systems," in *Proceedings of the 34th IEEE control and decision conference*, vol. 3, New Orleans, LA, USA, December 1995, pp. 2990–2995.

[54] A. Johansson, M. Bask, and T. Norlander, "Dynamic threshold generators for robust fault detection in linear systems with parameter uncertainty," *Automatica*, vol. 42, pp. 1095–1106, 2006.

Appendix A

Wind Turbine Bench Mark


Sensors
v_hub
Omega_r
Omega_g
tau_g
P_g
Beta_1
Beta_2
Beta_3
v_hub_m
Omega_r_m1
Omega_r_m2
Omega_g_m1
Omega_g_m2
tau_g_m
P_g_m
Beta_1_m1
Beta_1_m2
Beta_2_m1
Beta_2_m2
Beta_3_m1
Beta_3_m2
Controller
Beta_r
Tau_g_r
P_g
P_r
Omega_g
Constant
To Workspace
v_wind
Blade&Pitch System
Tau_r
Beta1
Beta2
Beta3
v_w
Omega_r
Beta_r
Wind model
avg_wind
v_hub
D_v_wind
From Workspace
Generator & Converter
Tau_g
Tau_g_r
Drive Train
Omega_r
Omega_g
Tau_r
Tau_g
Scope
Goto
From