Steady State Detection, Data Reconciliation, and Gross Error Detection: Development for Industrial Processes

by

Rocio del Pilar Moreno

Specialist in Mobile Communications,
Francisco Jose de Caldas District University, 2004
Bachelor in Electrical Engineering, UIS, 2001

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

In the Graduate Academic Unit of Electrical Engineering

THE UNIVERSITY OF NEW BRUNSWICK

January, 2010

Dedication

To God, for giving me the strength to complete this work.

To my mother, my grandmother and my brother, for their love, support, and for being the inspiration that motivates me to fight every day to achieve my goals.

To Pim, my loved partner. For his love, patience, support and for being my company during the good and hard times along this work.

Abstract

Measured data in chemical processes are subject to be corrupted by noise. Reliable data is very important to achieve a high-quality controlled process. There are three important aspects of data processing to improve the performance of the control system: Steady-State Detection (SSD), Data Reconciliation (DR) and Gross Error Detection (GED). This thesis research developed a steady state detection algorithm, extended and applied the Adaptive Nonlinear Dynamic Data Reconciliation (ANDDR) and Novel Gross Error Detection (NGED) approaches developed by Laylabadi and Taylor [1], and applied these contributions to the two-phase separator followed by a three-phase gravity treator model used in oil production facilities, derived by Sayda and Taylor [2]. It also developed a new hybrid approach to perform DR efficiently in complex processes.

Applying these algorithms to such a complex plant is a challenge, which allows the techniques to be tested in a realistic process, and at the same time, brings to the implementation several difficulties that were not faced in previous case studies.

Acknowledgements

I would like to express my most sincere gratitude to Dr. James H. Taylor. His wise advice, guidance, encouragement, and patience were fundamental and essential for the development of this work. Without his endless assistance and support, this thesis would have never been completed. This project is supported by Atlantic Canada Opportunities Agency (ACOA) under the Atlantic Innovation Fund (AIF) program. I gratefully acknowledge that support and the collaboration of the Cape Breton University (CBU), and the College of the North Atlantic (CNA).

I would also like to thank the faculty and staff of the Department of Electrical and Computer Engineering at University of New Brunswick for their patience, help and continuous encouragement. Specially I want to thank to Mrs. Denise Burke for her kindness and help during my studies.

Special thanks to Dr. Maira Omana for her continuous advice, support, encouragement, but most importantly for her friendship.

I extend my gratitude and appreciation to everybody who helped me during the journey of completing my Master degree: To my family, for the support and love; to my loved partner Pim, for being my strength in the difficult times, for his company, for his faith on me and for his love; to all the wonderful friends that made my study a happy, rewarding, and unforgettable experience.

Table of Contents

List of Tables

List of Figures

List of Symbols and Abbreviations

Symbols

A/D	Analog to digital
d	Difference between measurement and mean
D/A	Digital to analog
Δt	Sampling time
Fg_{out}	Gas molar outflow from the separator
Fg_1	Gas component flowing out of oil phase
Fg_2	Gas component dissolved in oil phase
Fh_1	Separated volumetric flow component of hydrocarbon fluid
Fh_2	Unseparated volumetric flow component of hydrocarbon fluid
F_{in}	Fluid injected into separator
Fo_{out}	Oil discharge flow
Fw_{out}	Water discharge flow
H	DR window size function
H_o	Null hypothesis for statistical hypothesis testing
$\hat{m}_k$	Mean estimated at sample k
Mw_g	Gas molecular weight
Mw_h	Hydrocarbon molecular weight
Mw_{in}	Mixture molecular weight
Mw_o	Oil molecular weight

Mw_w	Water molecular weight
N_{gas}	Number of gas moles in the gas phase
N_{oil}	Number of liquid moles in the oil phase
P	Total pressure of the vapor phase
Φ	Objective function
Ψ	Process dynamic constraints
R	Universal gas constant: 8.314475$(J/Kmol)$
SG_g	Gas specific gravity
SG_h	Hydrocarbon specific gravity
SG_{in}	Incoming mixture specific gravity
SG_o	Oil specific gravity
SG_w	Water specific gravity
σ	Standard deviation
$\hat{\sigma}_k$	Estimated standard deviation at sample k
T	SSD threshold
T	Absolute separator temperature
τ	Time constant
V_{gas}	Volume of gas
V_{oil}	Volume of oil
V_{sep}	Volume of the separator
V_{wat}	Volume of water
x	Mole fraction of gas into liquid phase
$\hat{y}_k$	k^{th} Estimate values
$\tilde{y}_k$	k^{th} Measurement values
Zg	Gas molar fraction
Zo	Oil molar fraction
Zw	Water molar fraction

Abbreviations

AI	Artificial intelligence
ANDDR	Adaptive nonlinear dynamic data reconciliation
CSTR	Continuously stirred tank reactor
DMI	Dynamic model identification
DR	Data reconciliation
FDIA	Fault detection identification and accomodation
GBN	Generalized binary noise
GED	Gross error detection
GEDR	Gross error detection and removal
GLR	Generalized likelihood ratio
GPS	Generalized parity space
ICAM	Intelligent control and asset management system
JCSTR	Jacketed continuously stirred tank reactor
MAS	Multi-agent system
MPI	Novel gross error detection
NGED	Signal to noise ratio
NLP	Nonlinear programming
PAWS	Petroleum application of wireless systems
PC	Principal component test
PCG	Preconditioned conjugate gradients
PEM	Prediction error/maximum likelihood method
RMA	Remote memory access
RMSE	Root mean square error
RPC	Remote procedure call
SSD	Steady-state detection
WLS	Weighted least squares

Chapter 1

Introduction

Data from real industrial and chemical processes are degraded by different types of noise. This is related to problems in the sensors such as incorrect calibration, low resolution, high-frequency pick-up, low-frequency pick-up, malfunctions, etc. Data corruption can also be due to errors in transmission and conversion, including A/D conversion and D/A conversion. These effects may also result in large discrepancies (gross errors). As a consequence, the laws of conservation of energy and mass may not be fulfilled, process dynamic behavior may not be matched, and trends may be obscured, producing mistakes in process diagnosis, identification and control. Reliable data is very important in order to achieve a high-quality controlled process. There are three important aspects of data processing to improve the performance of the control system: Steady-State Detection (SSD), Data Reconciliation (DR) and Gross Error Detection (GED).

DR is a technique used to adjust the measurements according to conservation laws and dynamic constraints. Several techniques have been developed, extending the scope to deal with concerns such as nonlinear behavior and dynamic data reconciliation. Liebman et al. [3] developed a Nonlinear Dynamic Data Reconciliation (NDDR) method and showed the advantage of using nonlinear programming (NLP) over conventional steady-state methods [3]. Based on this technique, Laylabadi and Taylor [1] [4] proposed an Adaptive Nonlinear Dynamic Data Reconciliation (ANDDR) algorithm and a novel GED approach, which includes the application to processes where the noise source has an unknown statistical model. The main difference between DR and other filtering techniques is that DR uses explicitly the process model constraints to find estimates of process variables by adjusting the measurements so that the estimates meet the constraints. Therefore the reconciled estimates are more precise than the measurements and, more importantly, are consistent with the relationships between process variables defined by the static and dynamic constraints.

Despite the substantial literature on such data processing methods showing their multiple advantages, there are not many application of these techniques on realistic large-scale industrial processes. There are significant obstacles to overcome, such as the unavailability of nonlinear mathematical models and the difficulty of running computationally intensive model-based algorithms in real time. This thesis is focused on the extension, implementation and assessment of the ANDDR and GED techniques applied to a three-phase gravity separator model used in oil production facilities. In addition to this, a multivariable steady-state detector algorithm has been developed. These two methods will be used as part of the Multi-agent System for Integrated Control and Asset Management of Petroleum Production Facilities [5].

1.1 Literature Review

1.1.1 Data Reconciliation

Kuehn and Davidson [6] were the first to address the problem of data reconciliation in 1961. They described the DR problem for steady-state chemical engineering processes. The method proposed involved the solution of an optimization problem that minimizes a weighted least-squares objective function of the error between measured and estimated values of the process variables under static material and energy balance constraints. Mah et al. (1976) [7], treated the general linear data reconciliation problem. The paper demonstrated that data reconciliation improves data accuracy (compared with standard filtering methods), especially when sufficient redundancy exists in the measurements.

In the study of dynamic linear processes, Kalman filtering [8] has been effectively used to smooth measurement data since its inception (see Gelb (1974)[9]). Stanley and Mah (1977) [10] were the first to tackle data reconciliation in dynamic processes using an extended Kalman filter. They used a simple random walk model to characterize the process dynamics. In 1990, Almasy [11] introduced a new method called *dynamic balancing,* which is based on the use of linear conservation equations to reconcile the measured states. In this approach only balance equations were utilized. Later Robertson et al. [12] presented a formulation of the dynamic data reconciliation problem as a special case of a more general moving-horizon state estimation method. State estimation problems have been often handled by filtering and moving-horizon optimization approaches [13].

The inclusion of nonlinear systems in the DR problem was first handled by Knepper and Gorman (1980) [14]. They used an iterative technique for parameter estimation in nonlinear regression. Jang et al. (1986) [15] made a comparison between Kalman filtering and nonlinear state and parameter estimation. The conclusion of this study was that using nonlinear programming (NLP) was superior in regard to response to changes in parameters and robustness in the presence of modeling errors and strong nonlinearities. In addition, Kalman filter approaches do not support the inclusion of variable limits or other inequality constraints. The cost of better performance for NLP approaches was longer computational time.

In 1992 Liebman et al. [3] developed a new approach for NDDR, which uses enhanced simultaneous optimization and solution techniques associated with a finite calculation horizon. Subsequently, Liebman and Edgar [16] demonstrated the advantage of using NLP techniques over conventional steady-state DR methods. They included variable limits and nonlinear algebraic constraints, improving the performance of the reconciliation. Based on the NDDR model developed by Liebman et al. [3]; Laylabadi and Taylor [1] proposed an Adaptive NDDR method that includes the application to processes with unknown statistical models.

While the dynamic data reconciliation problem has been widely studied, there are not many applications in real industrial scenarios, due to the difficulty of solving large nonlinear programs in real time. Romagnoli et al. (1996) [17] presented the application of steady-state data reconciliation to an industrial pyrolysis reactor. In this work, they used linear and nonlinear methods, concluding that the large computational time needed by the nonlinear method could not be justified. McBrayer et al. (1998) [18] reported the successful application of the NDDR algorithm developed by Liebman, to reconcile actual plant data from an Exxon Chemicals process. Gross error detection was not included in their approach. Placido and Loureiro (1998) [19] applied steady-state data reconciliation to measurements obtained from various units of a Brazilian ammonia plant. Soderstrom et al. (2000) [13] demonstrated the large scale application of NDDR for improving process monitoring.

There are several differences between the method and implementation presented in this thesis and the previous applications of DR to real models: In this work a new hybrid algorithm was implemented to make the DR process more efficient for application with complex models. The algorithm implemented in this thesis covers nonlinear, dynamic and steady state DR, and GED. Romagnoli et al. [17] did not use nonlinear methods (even though it was proven that NLP was more robust and efficient than conventional steady-state methods). In the applications presented by Romagnoli et al. [17] and Placido and Loureiro [19] just steady-state DR is performed, while the algorithm presented in this work is capable of achieving dynamic DR as well. The studies developed by McBrayer et al. [18] and Soderstrom et al. [13] implemented nonlinear dynamic data reconciliation but they do not include gross error detection in the methodology. The model used in the application presented by McBrayer et al. [18] was developed using several simplifications such as constant pressure and only one mass balance equation. Furthermore, the fact that the algorithm used in this thesis combines dynamic data reconciliation and gross error detection, it is important to stress the inclusion of the Adaptive NDDR + GED into an intelligent control system agent. So this method is applicable to steady state systems, dynamic systems, and systems with gross errors while it is able to exchange information and interact with other agents and a supervisory system.

1.1.2 Gross Error Detection

Gross errors are random or deterministic errors which have no relation with the true values. Initial approaches in DR studies usually assumed that the noise that affects the variables is randomly distributed with zero mean. However, gross errors can occur, as mentioned previously, and it is very important to detect them and remove them before or simultaneously with DR, in order to avoid corrupted adjustments. Almasy and Uhrin [20] studied the reasons for unexpected measurement data and they classified as gross error, the corrupted data obtained by malfunctioning instruments, measurement biases and process deficiencies.

In 1965, Ripps [21] pointed out the problem of identifying gross errors and its importance in DR. He used the procedure of measurement elimination as a test for gross error. In this method each measurement is deleted in turn. By eliminating a measurement, the corresponding variable becomes unmeasured and the objective function value will decrease. Ripps proposed that the gross error can be identified in that measurement whose deletion leads to the greatest reduction in the objective function value. This becomes one of the standard strategies in multiple nonsequential gross error identification. Several studies like Ripps [21], Reily and Carpani [22], Almasy and Sztano [23] and Madron et al. [24] proposed a global test for GED. The global test is based on statistical hypothesis testing, where the null hypothesis, H_0 (no gross error is present), is tested against the data set. Global tests can determine whether or not a data set contains gross errors but will not indicate precisely where. A detailed description of the global test can be found in [25]. A decade later Narasimhan and Mah [26] proposed the generalized likelihood ratio (GLR) test for detecting gross errors in steady-state processes. This test is based in the maximum likelihood ratio principle used in statistics. The formulation of this test requires a model of the process in the presence of a gross error. In 1995 Tong and Crowe [27] defined the principal component (PC) test. The PC test is a linear combination of the eigenvectors of the variance-covariance matrices of constraint residuals and of measurements adjustments; PC tests cannot identify the location of gross error.

Along with data reconciliation, methods to identify gross errors in dynamic systems are also emerging. For example, Soderstrom et al. [28] proposed a method to deal simultaneously with the problem of GED and model identification, together with data reconciliation; and Laylabadi and Taylor [1] [4] proposed a Novel Gross Error Detection technique (NGED), applicable to dynamic nonlinear processes with an unknown statistical model.

1.1.3 Steady-State Detection

Steady-State Detection (SSD), is an important step in control processes and is critical for process performance assessment and the application of other functionalities such as optimization. Several techniques such as Fault Detection Identification and Accommodation (FDIA) and Dynamic Model Identification (DMI) require the system to be in steady state in order to produce correct results. The majority of methods for SSD are based on calculating either the mean, variance or regression slope over a data window, and comparing them with results over the previous window applying statistical tests [29].

Narsimhan et al. proposed a statistical test [30] and a theory of evidence for the detection of changes in steady states [31]. These methods inspect successive time periods in which the variables are supposed to be in steady state, but this is an assumption difficult of satisfy in practice. In 1994 three solutions to the problem of identifying steady-state condition automatically were posted in a journal. The first technique, developed by Loar [32], suggested a moving mean and thresholds of $\pm 3\sigma$. This method is used to trigger control, but it cannot assure steady state. The second method, introduced by Alekman [33], compares the mean from a recent history to a "standard" based on an earlier history and then applies a t-statistic test to determine if the average changed. A problem for this approach is that a steady-state condition is not generally equivalent to the mean. The last method published by Jubien and Bihary [34] was based on calculating the measurement standard deviation over a moving window of recent data history. The measured standard deviation must be between an established threshold to declare steady-state condition. The success of this method relies on the ability to determine the process variables time period used for calculation. In 1995 Cao and Rhinehart [35] presented a modification to the F-test type of statistic in order to treat the data sequentially without the need of a time window. This was done by incorporating an exponentially weighted moving-average filter to calculate the average and variance by two different methods. Due to the filtering nature of the identifier, an amount of delay is present in the response of the steady-state identifier.

1.2 Objectives

The objectives of this thesis research are as follows:

1. Extend and implement the ANDDR approach on a two-phase separator followed by a three-phase gravity separator model used in oil production facilities [2].
2. Extend and apply the NGED method [1] [4] on the separator model [2].
3. Refine and test the performance of the ANDDR and novel GED package using a nonlinear model of a more realistic and complex chemical process.
4. Develop a new hybrid approach able to perform efficiently DR in complex models, and most importantly, to eliminate the previous requirement for a dynamic nonlinear process model.
5. Develop an algorithm capable of establishing when a multivariable system has reached the steady-state, and apply it to the separator model [2].
6. Implement the ANDDR + NGED and the steady-state algorithm as two compatible agents to work in conjunction with a smart supervisory system [5].

1.3 Contributions

The requirements demanded from control and automation techniques are continuously increasing due to the need for faster and more reliable results. That being said, the best control system performance can only be achieved by using accurate measurements. As a result, DR, gross error and steady-state detection have become crucial tools for data quality improvement in integrated control and asset management systems.

These three methods have been widely studied, but there are few applications to real industrial models, especially not to realistic nonlinear and dynamic processes. Implementing the ANDDR and novel GED techniques [1] on a real industrial process requires facing new challenges that are not present in research using ideal models, and thus necessitates finding solutions and tools to overcome the difficulties inherent to real industrial processes. The new hybrid approach solution is an important contribution to perform DR efficiently in systems with complex models. Modifying the algorithms in order to make them compatible with a multi-agent supervisory system [5] is another contribution that will facilitate future applications to industrial processes.

1.4 Thesis Outline

The Multi-agent System for integrated control and asset management of petroleum production facilities [36] is explained in chapter 2, to establish the context for the algorithm development in this thesis. Next, in chapter 3 the theory, methodology and results obtained for the steady-state detection agent are presented. The DR problem is described in chapter 4, as well as the solution adopted here and the results obtained. A new approach to tackle the data reconciliation problem is presented in chapter 5. Chapter 6 describes GED algorithm and illustrates its performance. Finally, in chapter 7 conclusions and future work are discussed. Appendix A describes how these contributions will be used in a multi-agent supervisory system. Appendix B presents a detailed description of the oil production facility model, which is the process used in this application. Appendix C shows the performance of the SSD algorithm, and Appendix D presents a comparison between DR and a low pass filter.

Chapter 2

ICAM System

2.1 Introduction

A major research project, PAWS (Petroleum Applications of Wireless Systems), is being pursued by several universities in Atlantic Canada for oil and gas applications. The UNB PAWS project objective is to develop a control and information management system. The overall project is divided in two areas: One, led by Cape Breton University (CBU), is focused on the wireless sensor network which reduces the utilization of data cables in offshore oil rigs, and the UNB portion is oriented to intelligent management and control of data and processes. For more information about PAWS project see [5].

To obtain accurate, reliable and efficient control in a modern process plant requires extensive supervisory monitoring and control. Several actions have to be executed, including steady-state detection, data reconciliation, fault detection, isolation and accommodation (FDIA), process model identification, and supervisory control. In order to maximize the efficiency of the process control, a multi-agent system (MAS), capable of integrating, supervising and managing all these tasks, had been designed and prototype built. This system, developed by the University of New Brunswick

(UNB) as part of the PAWS project, is the *intelligent control and asset management system* (ICAM system)[5] [37]. Implementing such a management system reduces maintenance and production costs, improves utilization of manufacturing equipment, enhances safety and improves product quality.

2.2 The ICAM System Prototype

The diagram shown in Figure 2.1 illustrates the simplified ICAM prototype [37] [5].

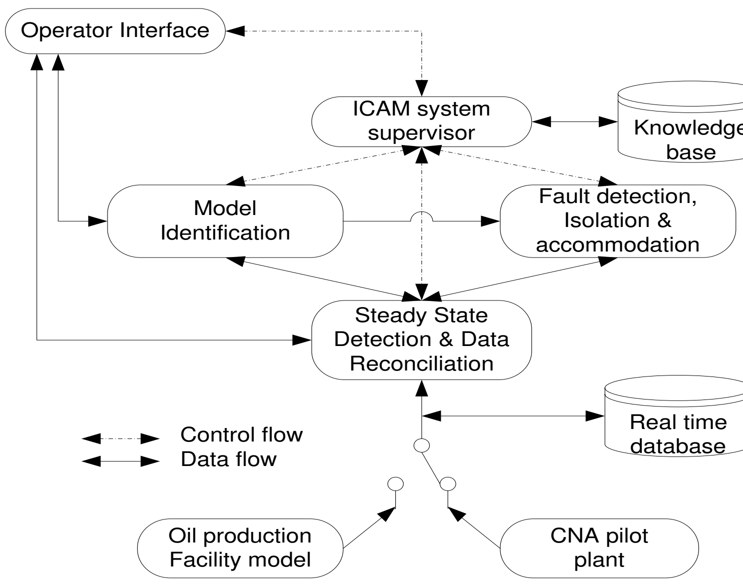


Figure 2.1: ICAM system prototype

Data are obtained in real time either from an external plant or from a simulation model. These data go to the statistical pre-processor and reconciliation block. This component is constituted by two different agents. The first one is the Steady-state Agent, which determines if the plant is either at steady or transient state, and the second is the Data Reconciliation Agent, which reduces the noise and removes outliers. Processed data are stored in a real-time database.

If there is no model available or if a significant change in the process operating point occurs, the Model Identification Agent is executed. This Agent uses generalized binary noise (GBN) signals as test signals to perturb the process inputs and to collect control relevant information about the process dynamics and its environment. Once the new model is obtained, the process model parameters are updated and loaded in the data reconciliation and FDIA Agents [38]. Subsequently, data are received by the FDIA Agent, which establishes if the system is being affected by a sensor or an actuator fault, classifies the type and size of fault, and accommodates the fault if this has an effect on a sensor. The FDIA Agent informs the supervisor if a fault has occurred in order to proceed with the appropriate actions [39].

The supervisor is alerted about every event that occurs; in this way it monitors, observes, and controls the system. An operator interface receives the data, and the information from the supervisor relative to the different agents. This allows the operator to take decisions according to the system status and requirements. The external plant for this particular project represents an oil production facility, which separates oil well fluids into crude oil, gas, and water. The plant itself is at the College of North Atlantic (CNA); however, for all research so far a realistic model of this plant [2] has been used. This process is explained in appendix B.

The ICAM system prototype was designed as the combination of three different layers:

- The Artificial Intelligence (AI) layer: This is the platform for the supervisory agent. It organizes and manages the reactive agents to obtain an optimum response.
- The middleware layer: This layer facilitates the communication between reactive agents and the supervisory agent.
- The reactive agents layer: It is composed by the data processing functions such as the FDIA Agent, Model Identification Agent, Data Reconciliation Agent, Steady-state Agent, Pilot Plant simulation, and, in the near future, Wireless Sensor Network Coordinator Agent [40].

2.2.1 The Artificial Intelligence Layer

The G2 real-time expert system shell is the platform for the implementation of the supervisory agent. The G2 codifies in its knowledge base the ICAM system internal and external behavior. The supervisory agent has an ontology that corresponds to the different reactive agents. The attributes and methods of each agent are represented by the agent technical characteristics and the agent's behavior respectively. The ICAM system can be defined as the logical connection among the different agents. There are two connections for each reactive agent, the first one is the MPI data exchange to share information with other agents, and the second is the G2 connection with the supervisory agent.

The ICAM internal and external behavior is coordinated by the Supervisory Agent. A rule-base design was established to achieve robust system performance. The event sequence which illustrates the rule-base design is described in section 2.3 [37].

2.2.2 The Middleware Layer

The communication between the reactive agents is performed using the remote memory access (RMA) communication approach, which is part of the message passing interface (MPI) communication library. This protocol supports two functionalities. The first is active target communication, where both transmitter and receiver are explicitly involved and data is moved from the memory of the former to the memory of the latter. The second is passive target communication, where only the origin process is explicitly involved in the transfer. The ICAM system was designed to use active target communication RMA in order to achieve high reliability. Four RMA data communication channels are used to transfer the different kind of information between agents: raw data, processed data, fault accommodation parameters and plant state space model.

The communication between the supervisory agent and the reactive agents is performed using the remote procedure call (RPC) paradigm, which allows achieving a looser connection with the supervisory agent. RPC is a client/server infrastructure that enhances the inter-operability, portability and flexibility of an application by allowing it to be distributed over multiple heterogeneous platforms. In the ICAM system prototype, the RPC communication approach was designed to make the G2 supervisory agent act as a client for the reactive agents which act as servers [37].

2.2.3 The Reactive Agents Layer

The ICAM system prototype is composed by four reactive agents:

1. Pilot Plant Agent: The Pilot Plant Agent (nonlinear simulation model) represents an oil production facility. The complete description of the pilot plant model is given in appendix B. The pilot plant is capable of running under four scenarios: the first one is the default scenario, which runs the model at its nominal operating point; the second allows changes in the set points; the third one applies perturbations to the plant for model identification, and the last one represents the plant when it is affected by faults in sensors or actuators.

2. Statistical Pre-processing Agent: This agent is actually composed by two different agents which are the focus of this thesis. The first agent is the Steady-state Agent which has to determine if the plant is in a steady or transient state and inform this state to the supervisor. The second one is the Data Reconciliation Agent. This agent is in charge of reducing the noise and removing undesired discrepancies such as outliers and missing data.

3. Model Identification Agent: This agent is executed only under two scenarios: if there is no linearized model for the process, or if there is a significant set-point change that makes the previously identified model invalid. First, the

excitation signals (generalized binary noise or GBN signals) are generated and subsequently they are applied as set-point variations to excite each control loop and generate the controller output (plant inputs) and the plant outputs. Then, the linearized state space model and the corresponding percentage of fitting for each output are calculated using the prediction error/maximum likelihood method (PEM) in MATLAB®for the reconciled inputs-outputs measurements [38].

4. Fault Detection, Isolation and Accommodation Agent: The FDIA Agent uses the generalized parity space (GPS) to create a set of directional residuals. From these residuals the faults can be detected and isolated. This agent is capable of establishing the size and type of fault, and which sensor or actuator is being affected. If the fault is present in a sensor, the algorithm is also able to accommodate the fault, by correcting the sensor data. For further description about the FDIA Agent refer to [38] [39].

2.3 The ICAM Event Sequence

The supervisory agent organizes the system's behavior using a rule-base design that coordinates the response to external changes and to operator interventions. Figure 2.2 shows a typical event sequence diagram. A brief explanation of the order of events is given to explain the context of the two agents developed in this thesis. For a complete description refer to [37].

The supervisory agent starts up all the reactive agents. The Steady-state Agent starts to work by doing a primary filtering of the measurements and checking and informing the supervisor constantly if the system is in a transient or steady state. The supervisor verifies the status of FDIA and NDDR agents to establish if they have a valid model or if they need a new one. Every time there is a significant set point

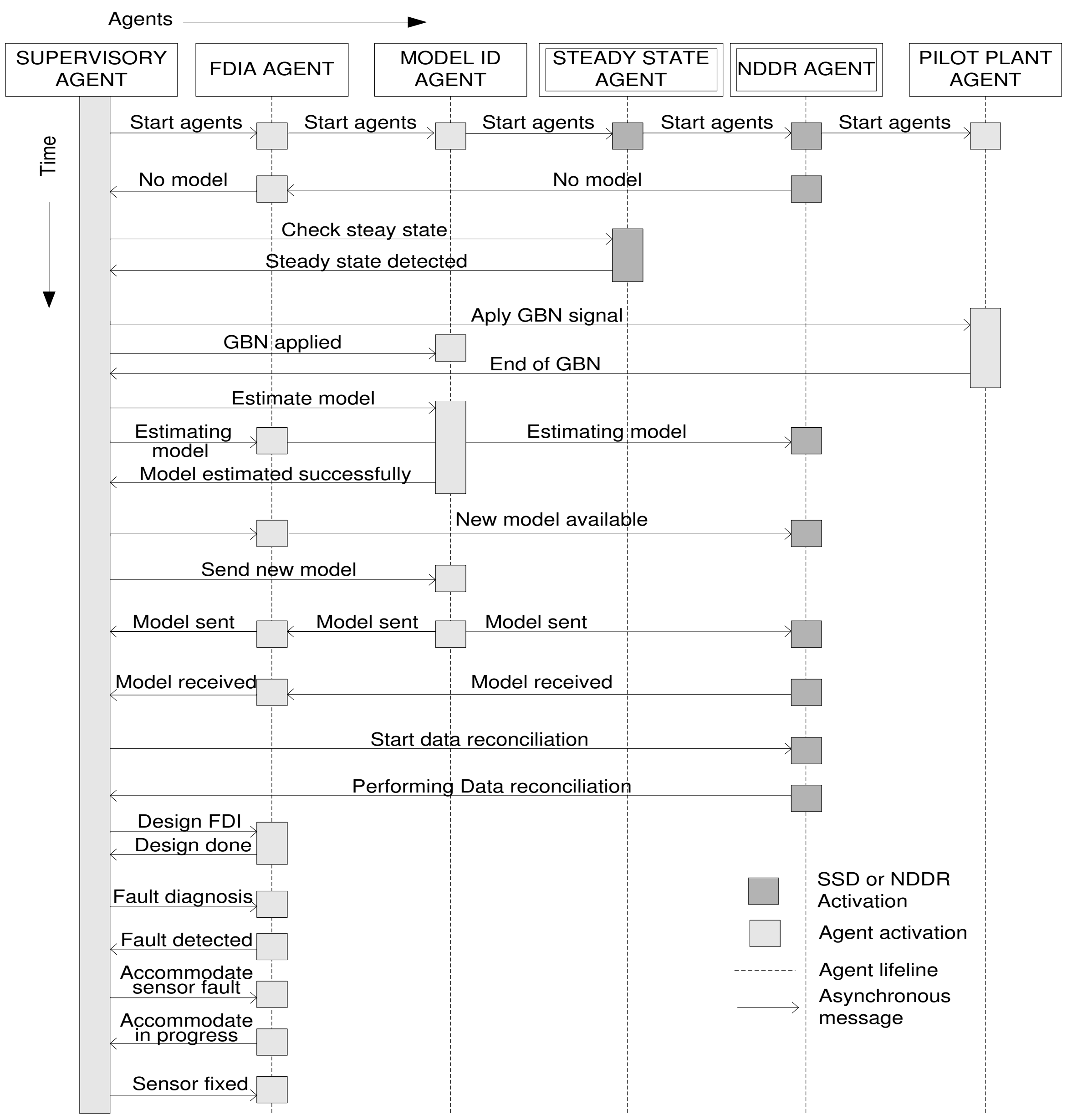


Figure 2.2: ICAM system prototype event sequence

change a new model is needed. The Model Identification Agent begins this process once the plant is in steady state, and reports to the supervisor when the new model is obtained. The supervisor sends the new model to the FDIA and NDDR agents to be updated. Once the new model is updated the NDDR Agent removes gross errors, reconciles the measurements and sends the cleaned data to the other agents. Using the reconciled data the FDIA Agent starts the process of fault diagnosis, fault detection and accommodation.

2.4 Steady State and NDDR Agent's Interaction within the Multi-agent System

In order to establish the importance of the two agents developed in this thesis within the multi-agent system, this section explains their relevance, the steps in the communication and the information interchanged.

There are three important stages in data processing: Steady-state detection, data reconciliation and gross error detection. Steady-state detection is essential due to the fact that several control and monitoring actions are developed for steady state processes. In particular, the model identification and the fault detection agents require the system to be in steady state for them to start to work and to provide a good performance. This is why the Steady-state Agent was developed and implemented as part of the ICAM system. Since measurements of process variables, such as volumes or pressures, are corrupted not only by normal noise but by gross errors as well, it is important to adjust the measurements and eliminate outliers in order to provide reliable data to perform proper control actions. Thus, DR and GED are applied together to improve accuracy of measured data.

The flow chart shown in figure A.1, in appendix A, illustrates the communication and data exchange process between the ANDDR + GED Agent, Steady-state Agent and other agents in the system: The steady state and data reconciliation agents are started up by the Supervisor Agent, and their G2 and MPI communication links are initialized. The first step is to take enough measurements to fill a window of H samples (H is the window size for data reconciliation). Afterwards, the standard deviation ($\hat{\sigma}$) and the mean ($\hat{m}$) of those measurements are estimated. These parameters are used to detect the presence of gross errors. If a gross error is present in the data, it is removed and replaced by the previous measurement (it is assumed that only isolated gross errors occur). Once the data is clean of outliers, the data reconciliation agent proceeds to reduce the noise by solving an optimization routine over the window. When data are free of outliers and noise is reduced, they are ready to be transmitted to other agents like the Supervisor Agent, FDIA Agent, Steady-state Agent, etc. A complete description and illustration of the Steady-state, Data Reconciliation and GED Agents is given in chapters 3, 4, and 6 respectively.

The following step for the system is to verify the state of the plant, that is, to establish if the plant is at either transient or steady state. To determine this, the Steady-state Agent uses a different data window than the one used for data reconciliation. When the window is full, a linear regression is performed and the attention is focused on the slope obtained from it. The slope is compared with a threshold and either if the conditions for steady-state are fulfilled or not, this information is passed to the supervisor. Once the steady-state is reached, other agents like Model Identification of FDIA can start to perform their tasks.

The supervisor must establish if large enough changes had occurred in the plant so that a new model is required. If a new model is necessary the supervisor starts the Model Identification Agent, which applies Generalized Binary Noise (GBN) to excite the plant and collect information about the process dynamics. Using this information the linearized state space model and the corresponding percentage of fitting for each output are calculated. The new model is sent and updated in the different agents.

Once the system is at steady-state and a suitable model is available, the FDIA Agent can be executed to establish if the system has been affected by a fault, to estimate the size and type of fault and to accommodate it. Every agent reports its status to the supervisor.

Chapter 3

Steady State Detection

Steady-state detection has become a very important step in process performance assessment, optimization, and control. There are several techniques for data processing and analyzing, used in chemical and industrial plants such as process optimization, fault detection and accommodation, model identification, etc. Most of these techniques require the system to be in steady-state in order to obtain optimal performance. For the case of the multi-agent system proposed in [5], the model identification and fault detection, isolation and accommodation agents require the system to be at steady state before they can start working.

This thesis presents a method for steady state detection based on linear regression over a moving data window. A description of the algorithm is given, followed by results obtained when applying this method to the pilot plant model.

3.1 SSD Algorithm

The majority of steady-state determination approaches work based upon statistical tests on the data. These strategies may involve calculating the average for a moving data window and comparing the current window average with the previous one, using a threshold of ± 3 standard deviation. Another method is based on obtaining two variances for the same set of data, using two different techniques: the variance can be calculated conventionally as the mean-square-deviation from the average, and it can also be calculated from the mean of squared differences of successive data. The ratio of these two variances is estimated and it has to be close to unity while the system is at steady-state or much larger than unity for unsteady-state or transient regimes.

Several approaches were studied in order to find an algorithm that offers good performance using the data from the pilot plant model. The first approach was based on the amount of change of the signal at every point, reflected in the measurement's derivative. Assuming y represents the signal's value, the derivative at current sample k is calculated as shown in equation 3.1.

$$m(k) = \frac{y_k - y_{k-1}}{t_k - t_{k-1}} \tag{3.1}$$

A test is performed on this parameter in order to detect steady-state. This method works well for all variables in a free noise measurements scenario. Unfortunately this is not realistic, given that in industry and more specifically in the PAWS application the variables are affected by different noise sources.

The second attempt was based on the statistics of the variables. This method calculates the difference between the current measurement and the average of previous data. Subsequently this difference is compared with the standard deviation. The method was successfully tested in different signals with noise, unfortunately it was not suitable for the pilot plant model data. As has been mentioned previously, the pilot plant is a complex system, and the nonlinear model developed by Sayda and Taylor [2] has some constraints due to the highly nonlinear behavior of the plant. Some restrictions are, for example, the variation in the setpoint and the amount of noise that the system can tolerate before it goes out of the safe operation zone. There are scenarios where the standard deviation of the noise may be greater than or close to the setpoint change. This situation makes it difficult to find a statistical pattern which helps to determine steady-state.

The method finally adopted in this thesis performs a least square linear regression over a moving data window. The purpose of this is to find the equation of the best-fitting line (in a least squares sense), for a set of data, and to analyze the rate-of-change of the line reflected in the slope. The equation of the line obtained is:

$$y_i = mx_i + b \tag{3.2}$$

where b is the y-intercept and m is the slope. Given that the method is going to be applied in the pilot plant which is a complex multi-variable process, every output is analyzed separately and when all the variables fulfill the condition for steady-state, the system is declared to be in steady-state.

A moving data window approach is used for this algorithm. Although the concept is similar to the moving window used in data reconciliation, the size of the window is different. The criteria for choosing this parameter depends upon the time constant of the variable, unlike in DR where the size of the window depends upon the sampling time. The advantage of using a data window is that it reduces the need to store data and the computational time.

The pilot plant model has five output variables. Every variable has a different time constant, and the range of variation between variables is wide. This is why every signal is evaluated independently using a different window size. The following table shows the time constant and the window size for the output variables in the pilot plant model.

Variable	τ (sec.)	Window Size (Samples)
Separator Volume ($V_{sep-liq}$)	23.44	200
Separator Pressure ($P_{sep-vap}$)	2.76	30
Treator Water Volume ($V_{treat-wat}$)	10.2	60
Treator Oil Volume ($V_{treat-oil}$)	3.6	40
Treator Pressure ($P_{treat-vap}$)	0.3082	20

Table 3.1: Time constants - Pilot plant Output Variables

Once enough data is obtained to fill the window, the linear regression is performed and its slope (m) is compared with a threshold. If the slope is smaller than the threshold for several samples (D samples), steady-state can be confirmed. Figure 3.1 illustrate the concept of the method adopted to detect steady-state: The figure shows the volume on the separator when a setpoint change of 10% its nominal operating value is applied at time $t = 0\ sec$. The noise standard deviation is 1% of the nominal operating value. When the signal is in the transient-state, the slope of the line is large (m1). At the maximum point the slope is small (m2), but this condition changes in a few samples. The closer the signal is to steady-state, the smaller the slope is (m3) and this condition continues.

The threshold (T) is not a constant but a function which depends upon the set point change, SP, and the standard deviation of the noise, both of which are assumed to be known. The calculation of this parameter was accomplished by doing several experiments, running the algorithm for different combinations of set point change and σ, and performing a multiple regression to fit the different outcomes of the tests. The equation for the threshold is:

$$T = a_0 + a_1\sigma + a_2 SP \tag{3.3}$$

where $a_0,\ a_1\ and\ a_2$ are the coefficients for the threshold model.

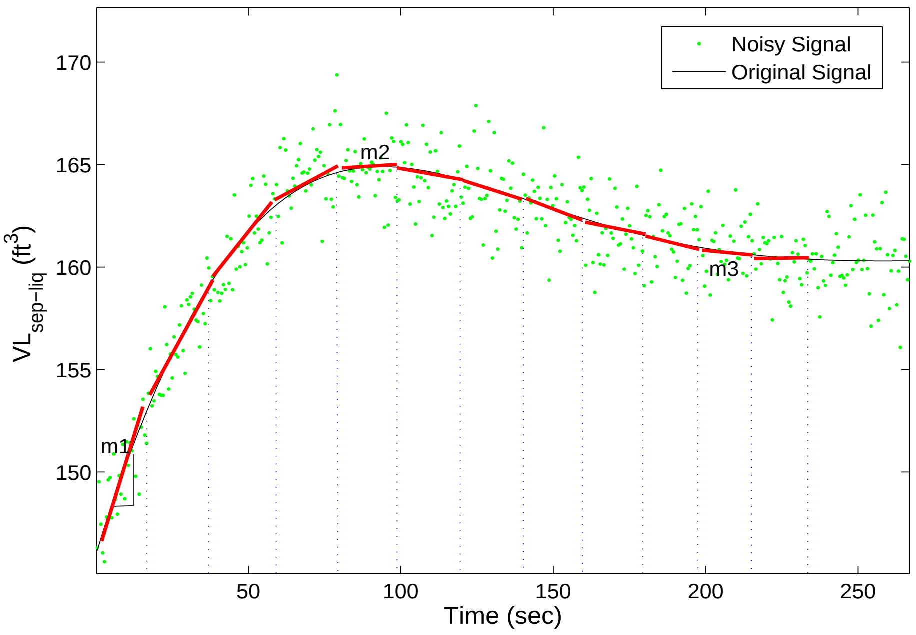


Figure 3.1: Steady-state Detection examples

Figure 3.2 shows a schematic flow-chart of the Steady-state Detection process.

3.2 SSD Algorithm Results

This section shows the results of the SSD algorithm when it is applied to measurements obtained by simulation, using the pilot plant model explained in Appendix B. Measurements were simulated and noise was added. The noise is assumed to be Gaussian with zero mean. The time step used is 0.15 seconds.

The setpoint change and the standard deviation of the noise are always expressed as a percentage of the nominal operating value for every variable. Table 3.2 shows the nominal values for the different inputs and outputs involved in the pilot plant model.

Figure 3.3 shows the response of the SSD algorithm for the volume in the separator $V_{sep-liq}$. The added noise has a standard deviation of 1% of the nominal set-point. In the simulation, the plant is working at its nominal operating point for the first $100sec$.

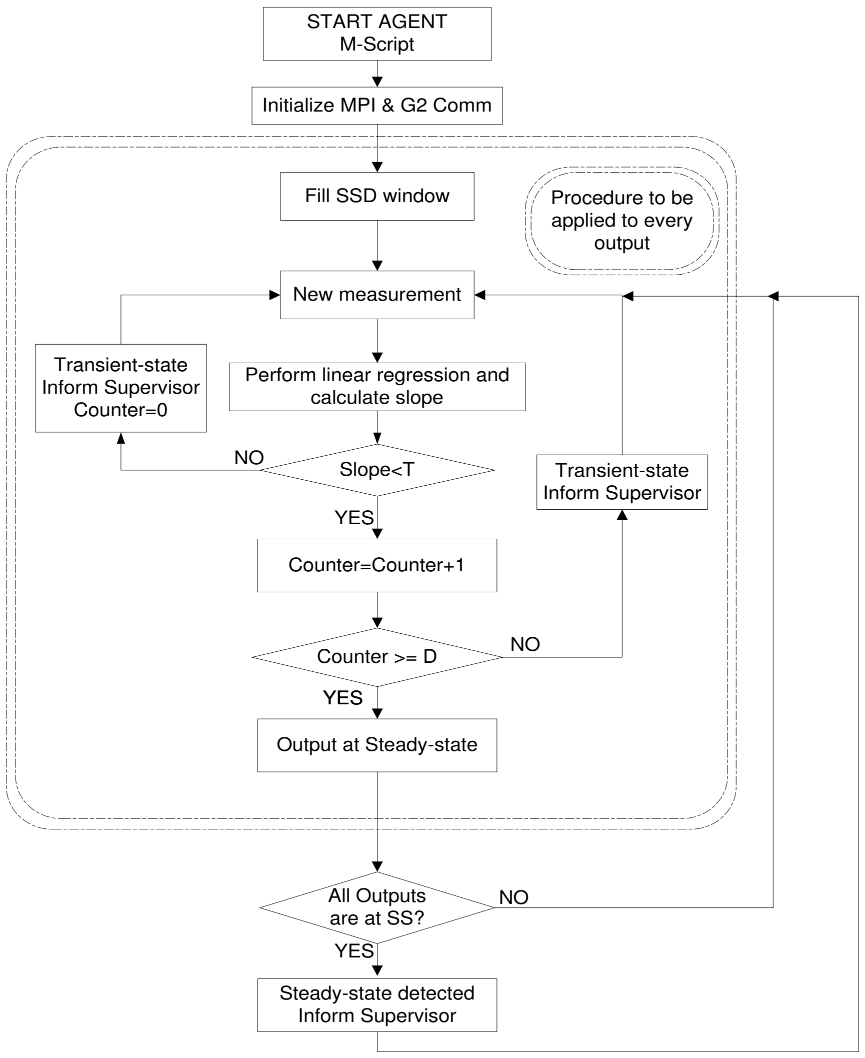


Figure 3.2: Steady-state Detection flow chart

At that time a set-point change of 5% of the nominal operating value is applied. The upper plot displays the linear regression slope and the lower shows the noisy signal and the original signal without noise. In both plots is possible to observe the SSD flag, which informs the supervisor about the state of the plant. If the SSD flag is zero, it means the signal is in unsteady-state, and if it is at the high level it means the variable reached steady-state. The high value of the steady-state flag sent to the supervisor is unity. However, in the following figures that value is modified in order to make the flag comparable with the associated variable.

Variable	Nominal operating point	Units
$V_{sep-liq}$	146.1	ft^3
$P_{sep-vap}$	625	PSI
$V_{treat-wat}$	77.48525	ft^3
$V_{treat-oil}$	46.49115	ft^3
$P_{treat-vap}$	200	PSI
$Fout_{sep-liq}$	20.31	$moles/sec$
$Fout_{sep-vap}$	5.01002	$moles/sec$
$Fout_{treat-wat}$	5.08198	$moles/sec$
$Fout_{treat-oil}$	2.0013	$moles/sec$
$Fout_{treat-vap}$	0.687	$moles/sec$

Table 3.2: Nominal operating point values - Pilot plant Variables

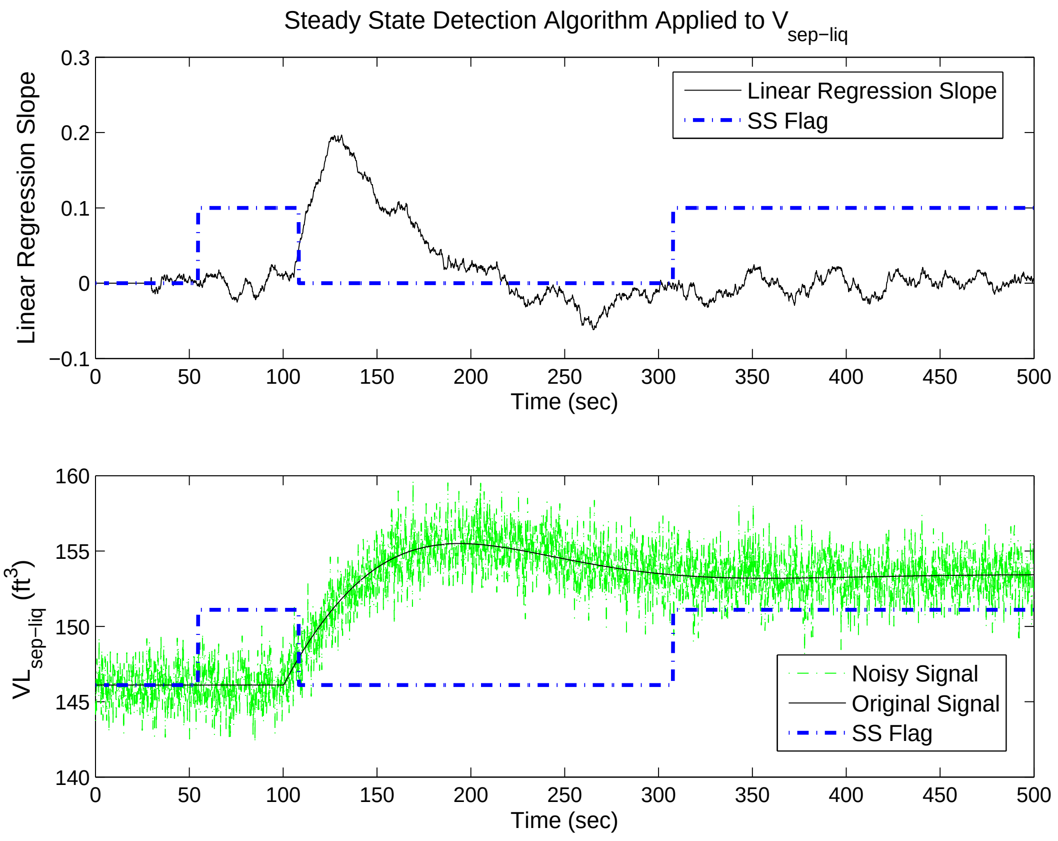


Figure 3.3: Steady-state Detection on V_{sep}

Figure 3.4 shows the results after applying the SSD algorithm to all the input and output signals involved in the pilot plant model. When the algorithm is applied to the complete plant there are two different kind of flags, an individual flag for every variable, and a general flag which informs the state for the complete system. For this case the plant starts at the nominal operating point. A set point change of 5% the nominal operating point value is applied to all variables at different times. At time $t = 80$ *sec.* a set point change is applied to $V_{sep-liq}$, subsequently at time $t = 400$ *sec.* is applied to $P_{sep-vap}$, at $t = 450$ *sec.* is applied to $V_{treat-wat}$, at $t = 550$ *sec.* is applied to $V_{treat-oil}$, and finally at $t = 650$ *sec.* the set point change is applied to $P_{treat-vap}$. The noise added to the signal has a standard deviation of 1%. These figures illustrate the different signals with their characteristics. It is observed that there are fast signals such as the pressure in the separator ($P_{sep-vap}$) and the pressure in the treator ($P_{treat-vap}$), which take shorter times to reach steady-state. On the contrary, the volume of the liquid in the separator ($V_{sep-liq}$) is significantly slower compared with all other variables and thus is the one that takes longer to reach steady-state. It is also shown that the algorithm is capable of establishing steady-state individually and for the complete system. Appendix C shows the successful results of the SSD algorithm for two other scenarios, confirming the accuracy of the SSD algorithm.

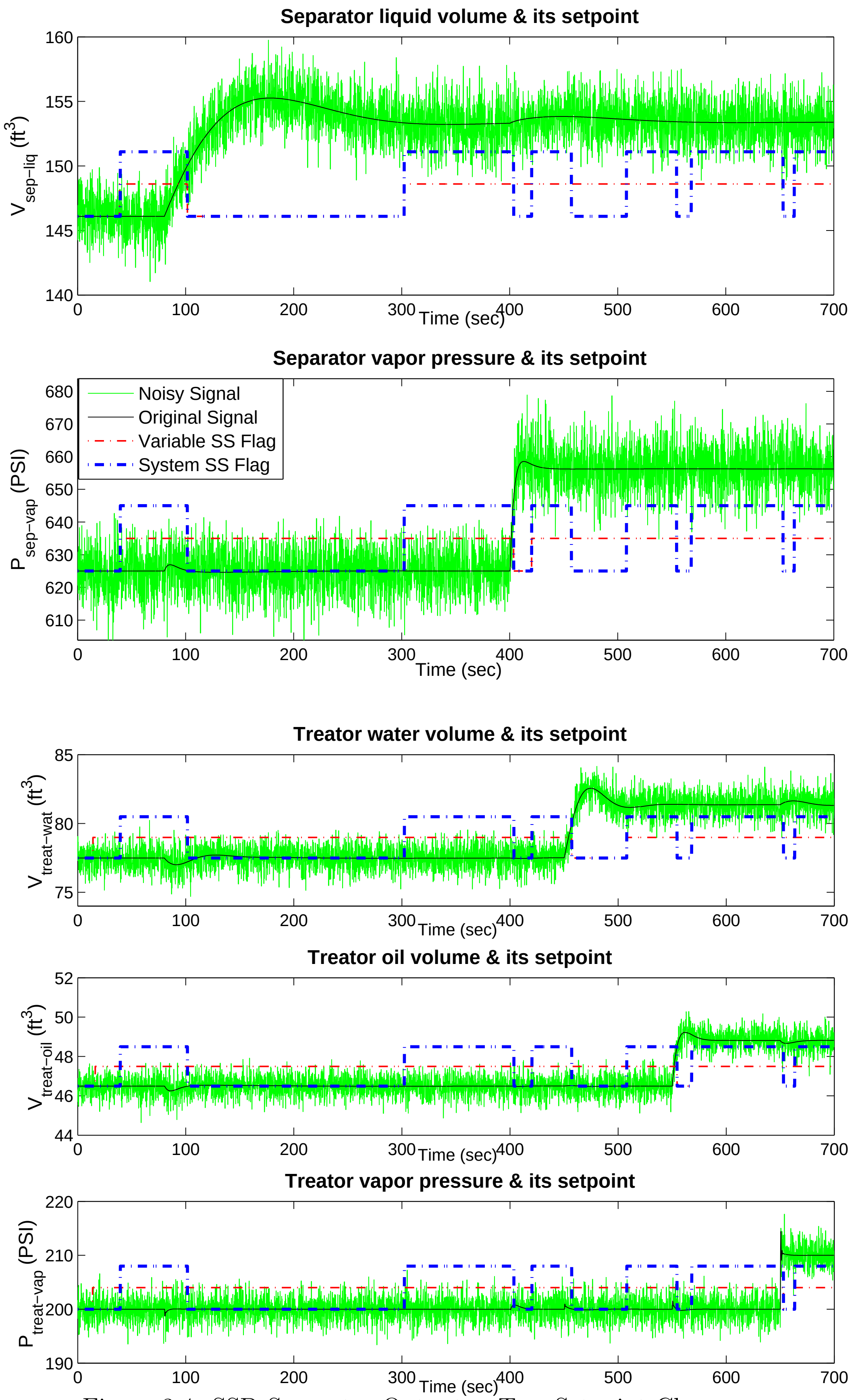


Figure 3.4: SSD Separator Outputs - Two Setpoint Changes

Chapter 4

Nonlinear Dynamic Data Reconciliation (NDDR)

Modern chemical plants, petrochemical processes and refineries, work by measuring and controlling several variables such as flow rates, temperatures, pressures, levels, compositions, etc. Sensed values of these variables are subject to be corrupted by random and systematic errors. Due to these errors, the relationship between the inputs and outputs of a system may not match with the process conservation laws. As it was explained in chapter 1, DR improves the accuracy of process data by adjusting the measured values so that they satisfy the process constraints.

Based on the method used by Laylabadi and Taylor [1], this thesis implements, refines, and assesses the ANDDR and GED techniques by applying them to the PAWS pilot plant model. The general formulation for the NDDR problem introduced by Liebman et al. [3] is discussed in this chapter, as well as the solution adopted to tackle the problem while implementing NDDR for a more complex model. Results of the basic NDDR algorithm in the pilot plant model are presented.

4.1 NDDR Formulation

The general NDDR formulation can be expressed as follows [3]:

$$\min_{\hat{y}(t)} \Phi[\tilde{y}, \hat{y}(t); \sigma], \tag{4.1}$$

subject to

$$\Psi(\frac{d\hat{y}(t)}{dt}, \hat{y}) = 0, \tag{4.2}$$

$$h[\hat{y}(t)] = 0, \tag{4.3}$$

$$g[\hat{y}(t)] \geq 0, \tag{4.4}$$

where

$\hat{y}(t)$	=	estimated (reconciled) measurements,
$\tilde{y}$	=	corrupted measurements,
Φ	=	objective function,
σ	=	measurement noise standard deviations,
Ψ	=	process dynamic constraints,
h	=	energy and/or material balance constraints,
g	=	process variable limits.

The lengths of $\hat{y}(t)$, $\tilde{y}$ and σ are equal to the total number of variables (states and inputs), i.e., $y = [x \mid u]^T$. Most of the applications use weighted least-squares (WLS) as the objective function in equation (4.1). The dynamic constraint in equation (4.2) is usually that the process differential equation must be satisfied, i.e., $\hat{y}$ is adjusted until the difference between integrating the system differential equation over a data window and the measurements $\tilde{y}$ over the window is minimized in the mean-square sense.

4.1.1 Solution Strategy

There are two important strategies adopted to facilitate the solution of the general NDDR problem: using a moving horizon data window and a process of discretization.

4.1.1.1 Moving Horizon Window

Liebman et al. [3] proposed a moving time window approach in order to decrease the size of the optimization problem. If t_c is defined as the present time, the *history horizon* is established from $t_c - (H - 1)\Delta t$ to t_c, with Δt as the time step size. It is important to choose an appropriate horizon length H. If H is too small, the information available may not be enough to perform a good reconciliation, but if it is too large, the NLP problem can become excessively large. The steps for NDDR can be summarized as:

1. Acquire process measurements at time $t = t_c$
2. Minimize Φ over the window $(t_c - (H - 1)\Delta t \leq t \leq t_c)$
3. Save $\hat{y}$ at time t_c as the reconciled signal for online control purposes
4. Repeat at the next step, t_{c+1}

The advantages of the moving window approach are:

- It reduces the size of the NLP problem.
- It does not require keeping all the previous information but just the size of the window, decreasing the data storage and computation requirements.
- Since the window has a finite length, information collected previous to the window does not affect the current estimation; this is important if unmodeled changes happen.
- The only tuning parameter for the NDDR algorithm is the size of the history horizon, H.

4.1.1.2 Discretization

The dynamic constraint, equation (4.2), needs to be discretized in order to solve the NLP problem defined by equations (4.1) to (4.4). To achieve this, the differential equations $\dot{x} = f(y)$ are solved numerically over the window using the Euler algorithm with a fixed step-size equal to the sampling time. The new NLP problem based on the discretized model and a WLS objective function can be rewritten as:

$$\min_{\hat{y}} \sum_{i=0}^{ni+ns} \eta_i \sum_{j=c-H}^{c} (\frac{\tilde{y}_{ij} - \hat{y}_{ij}}{\sigma_i})^2, \tag{4.5}$$

subject to:

$$\Psi(\frac{d\hat{y}}{dt}, \hat{y}) = 0, \tag{4.6}$$

$$h(\hat{y}) = 0, \tag{4.7}$$

$$g(\hat{y}) \geq 0, \tag{4.8}$$

where $\Psi(\frac{d\hat{y}}{dt})$, $h(\hat{y})$ and $g(\hat{y})$ correspond to the constraints obtained through discretization, η is a vector of weights, n_i is the number of inputs and n_s is the number of states; in order to maintain the maximum-likelihood nature of the estimation scheme, the weights η_i are all equal. If $\dot{x} = f(y)$ is a physics-based nonlinear model then the material and/or energy balance conditions are satisfied and $h(\hat{y}) = 0$ is not required. The inequality constraints may include limits on process variables, for example the separator input and output flows (FT in figure B.3) cannot be negative.

4.2 NDDR Results

The basic NDDR algorithm was implemented on the pilot plant model in order to assess the performance of this method in a large scale, realistic model, and as a first step to develop an agent capable of working within the ICAM system. For this test, five inputs and five outputs are being estimated. True values were obtained by simulating the nonlinear model at a time step of $\Delta t = 0.15$ *sec.*, and measurements were created by adding Gaussian noise to the true values; they were assumed gross error free. The noise added to the simulated data was Gaussian with zero mean, and it has a standard deviation of 1% of the nominal operating point value of the corresponding variable. For this first test, there is no change in the setpoint, all the variables are at their nominal operating point value. The window size, H was set to 10.

All the experiments shown in this thesis were performed in a DELL computer with the following specifications: INTEL® Core^{TM}2 Duo CPU, E8400 @3.00 Ghz. 2.99Ghz, 3.21 GB of RAM. The programs were performed in MATLAB®version 7.6.0 324 (R2008a).

4.2.1 Data Reconciliation Results Using `fminsearch`

The first attempt to implement the NDDR was very similar to the approach used by Laylabadi and Taylor [1]. The optimization was executed using the unconstrained nonlinear method `fminsearch`. This is a direct search method that does not use numerical or analytic gradients. Using this method on the separator model consumes a large amount of computation time. The reason for such a slow performance is because the optimization routine can not find a minimum, and it keeps iterating until the maximum number of iterations or maximum number of evaluations allowed are reached (200 times the number of variables). The conditions used in this test are:

- Window size H=10 samples.
- Initial guesses for optimization algorithm: Previous estimates.
- scaling: all the weights $\eta_i = 1$.

The results for this trial are shown in Figure 4.1. It can be observed that the estimates appeared to be significantly smoother than the corresponding measurements. The noise in the reconciled estimates was compared to the noise in the measurements over the complete running time by evaluating sample statistics of the differences between the measurements and the true values and the estimates and the true values. Table 4.1 provides a quantitative summary of the results for this test. The table shows the standard deviations for the noise on the measurements and for the noise on the reconciled estimates, and the percentage of reduction obtained. The root mean square error (RMSE) for the measurements and for the estimates and its percentage of reduction are presented as well. All estimate noise deviations and RMSE were significantly smaller than the corresponding measurement.

Even though the algorithm is capable of obtaining estimates with less noise, it takes an excessive large computation time to carry out this task. The algorithm was run for a time span of $t_f = 20.1$ *sec.*, and the computation time used to complete the routine was $t_{comp} = 21,094$ *sec.* (approx. 1,049 times real time). This large computation time is due to the fact that the pilot plant model is extremely complex, with an optimization problem needing to be solved to balance oil and water separation at each time step. The small sampling time is also a factor; it is based on the rapid gas pressure dynamics.

Variable		Measurem. Std. Dev.	Estimate Std. Dev.	% σ Reduction	Measurem. RMS error	Estimate RMS error	% error Reduction
Inputs	$Fout_{sep-liq}$	0.2042	0.0394	80.67	0.2042	0.0717	64.88
	$Fout_{sep-vap}$	0.0508	0.0128	74.69	0.0513	0.0221	56.88
	$Fout_{treat-wat}$	0.0492	0.0071	85.39	0.0490	0.0175	64.17
	$Fout_{treat-oil}$	0.0182	0.0034	81.24	0.0182	0.0084	53.57
	$Fout_{treat-vap}$	0.0072	0.0018	73.94	0.0071	0.0034	51.78
Outputs	$V_{sep-liq}$	1.3231	0.3155	76.15	1.3243	0.5235	60.46
	$P_{sep-vap}$	5.3857	1.6565	69.24	5.3699	2.0980	60.92
	$V_{treat-wat}$	0.7832	0.2425	69.02	0.7813	0.2659	65.96
	$V_{treat-oil}$	0.4591	0.1383	69.86	0.4578	0.1959	57.20
	$P_{treat-vap}$	1.9807	0.7026	64.52	1.9738	0.9950	49.58
Average		1.0261	0.3120	74.47	1.0237	0.4202	58.54

Table 4.1: NDDR first approach results

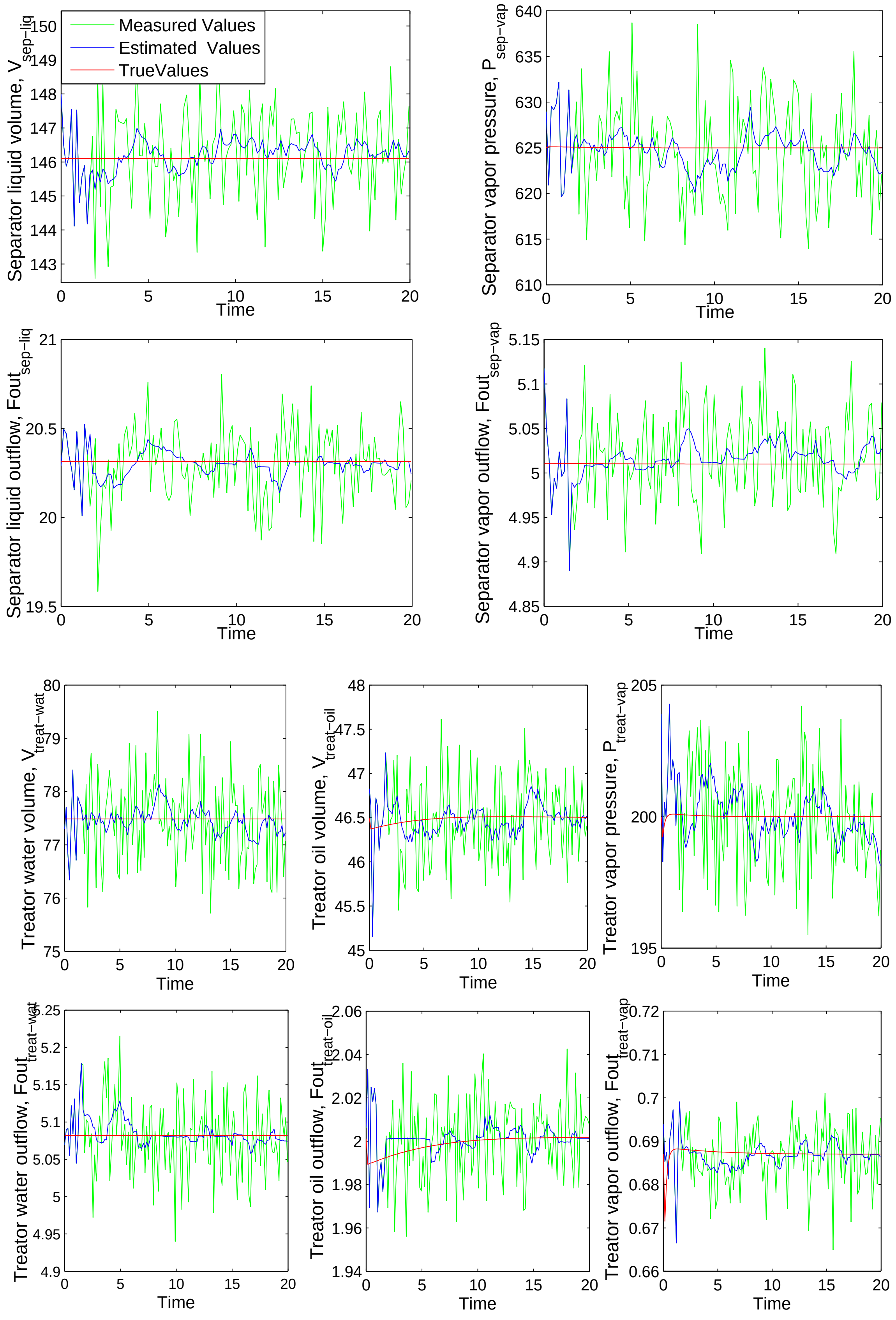


Figure 4.1: First implementation NDDR on Pilot plant model

4.2.2 Data Reconciliation Results Using `fminunc`

Although the results of the NDDR algorithm show a good performance in noise and RMSE reduction, the computation time is an important issue. The NDDR algorithm is going to be implemented as an agent which is part of the ICAM system, and in the future, the complete system is going to be executed in real time, therefore the NDDR algorithm needs to drastically improve the computation time.

The first step to reduce the computation time was to analyze the type of optimization routine used. As it was mentioned before, the previous test was implemented with the same method used by Laylabadi and Taylor [1] in order to establish the behavior of that approach in a more complex model. The minimization function used was `fminsearch`. In n dimensions, this method evaluates the objective function over a polytope (a simplex) of $n + 1$ points in the parameter space (in two dimensions, the simplex is a triangle). At each iteration, `fminsearch` computes the objective function at the points of the simplex, deletes the point with the highest (worst) objective function value, and replaces it by a new point giving a new simplex. Where appropriate, the simplex can shrink or grow in size. This is analogous to flopping a triangle around the parameter space until it finds a minimum. `fminsearch` stops when the objective function is the same (within some tolerance) in all points of the simplex, or when the size of the simplex is less than the specified tolerance, or when the iteration limit is reached. `fminsearch` requires no gradient information and can handle function discontinuities. The disadvantage of `fminsearch` is that it converges very slowly to the solution, especially for searches of three or more parameters [41]. The pilot plant model includes five inputs and five outputs, for a total of ten parameters to be estimated, this makes the function `fminsearch` not suitable for the problem addressed in this thesis.

The minimization function `fminunc`, which is an efficient large-scale algorithm, was thus selected to perform the optimization. This algorithm is a subspace trust-region method and is based on the interior-reflective Newton method explained in [42]. Each iteration involves the approximate solution of a large linear system using the method of preconditioned conjugate gradients (PCG). The idea of this algorithm is to form a linear approximation to the problem and solve it. This determines a direction to search along, and it predicts a step length in that direction. This is why the gradient is required in this function. For a complete description of `fminunc`, the trust-region method, and the PCG method, please refer to [41].

The NDDR was implemented using the large-scale optimization algorithm and it was tested in the same scenario as in section 4.2.1. The size of the window H is 10, the initial guesses used are the previous estimates, the scaling is equal for all the variables and is set to 1. The step time is $\Delta t = 0.15$ *sec.*, and the elapsed time is $t_f = 20$ *sec.*

The results are shown in figure 4.2. Table 4.2 shows the quantitative results for this test as well as a comparison between the results obtained with `fminsearch` and `fminunc`. An important reduction in the computation time was observed: using `fminsearch` the computation time was $t_{comp} = 21,094$ *sec.* (1,049 times the running time) and using `fminunc` the computation time was reduced to $t_{comp} = 1,368$ *sec.* (68 times the running time). Even though this time is still large, it is a big improvement.

The noise reduction achieved using `fminunc` is smaller that the reduction obtained using `fminsearch`, but still it is significant. The amounts of RMSE reduction are very similar between the two approaches. This may be due to the fact that the strategies for finding solutions are different for each algorithm, which means that the path of iterations can be completely distinct, so the iterations may find different solutions. `fminsearch` tends to be slower, especially for larger problems but it may be more robust to handle some problems such as derivative discontinuities.

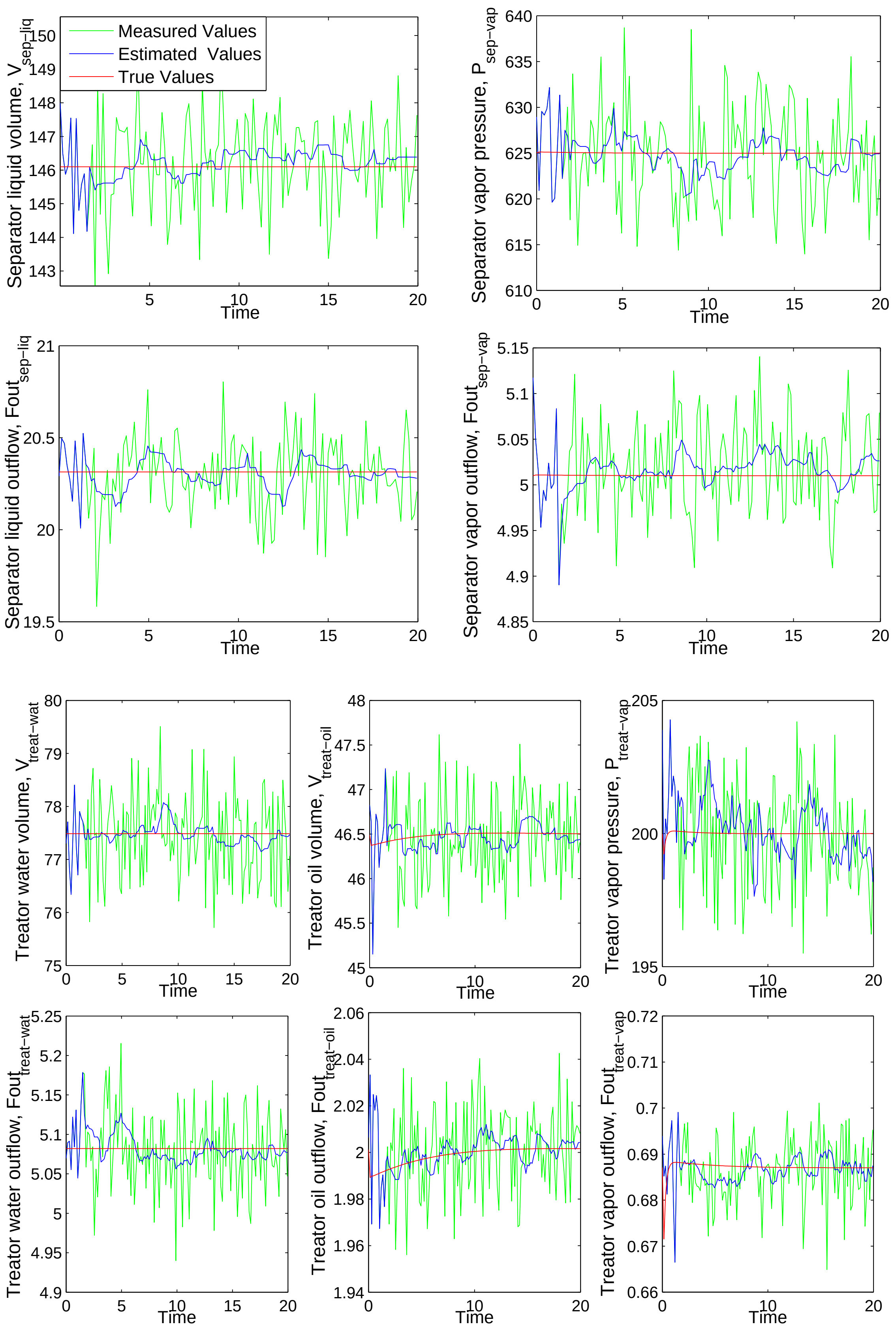


Figure 4.2: NDDR results Using `fminunc`

	Variable	% Noise reduction		% RMSE reduction	
		`fminsearch`	`fminunc`	`fminsearch`	`fminunc`
Inputs	$Fout_{sep-liq}$	80.68	65.62	64.88	60.21
	$Fout_{sep-vap}$	74.69	57.49	56.88	55.60
	$Fout_{treat-wat}$	85.40	59.01	64.17	61.59
	$Fout_{treat-oil}$	81.25	74.63	53.57	54.10
	$Fout_{treat-vap}$	73.94	63.60	51.78	50.14
Outputs	$V_{sep-liq}$	76.15	72.35	60.47	62.75
	$P_{sep-vap}$	69.24	69.49	60.93	61.72
	$V_{treat-wat}$	69.03	80.24	65.97	70.33
	$V_{treat-oil}$	69.87	70.52	57.21	60.39
	$P_{treat-vap}$	64.52	54.59	49.59	45.25
Average		74.48	66.72	58.54	58.21

Table 4.2: Comparison NDDR performance using `fminsearch` and `fminunc`

Even though the results show significant noise reduction in a shorter time, the time consumed is still too long. To enhance the overall performance of the NDDR algorithm, several options were tested. These options included modifying the size of the horizon window, trying different initial guesses for the optimization routine, and using scaling. Data used for the different tests are the same as described previously, but the algorithms were executed for an elapsed time of ($t_f = 10$ *sec.*), to shorten run times. The results and conclusions for these tests are presented in the following sections.

4.2.3 Modifying Window Size *H*

One of the advantages of using a history horizon window as part of the solution strategy for NDDR is that the length of the window acts as a tuning parameter for the performance of the data reconciliation scheme. For that reason, the size of the window (H) was the first parameter value to be investigated. Increasing H would provide more information for the optimization algorithm and it would yield more noise reduction, although the optimization would take longer. Therefore it is necessary to find a balance between time consumed and noise reduction achieved.

Tables 4.3 and 4.4 show the percentage of noise reduction and percentage of RMSE reduction respectively when the NDDR algorithm is executed using different values for H. The running time for the test is $t_f = 10.05$ *sec.* It can be observed that the time consumed to execute the algorithm increases proportionally to value of H. Regarding noise reduction and RMSE reduction, the results show that the larger H is, the smoother the estimates are, which is the expected behavior. The level of noise reduction and RMSE reduction increase substantially with H, but only up to a certain value (in this case approximately $H = 16$). For larger values of H, the improvement in the estimates is not significant. The level of noise reduction and RMSE reduction tend to stabilize for large values of H, while the time consumed continues increasing. Figure 4.3 shows the relation between size of window and percentage of noise and RMSE reduction for $Fout_{treat-vap}$. It can be observed the effect explained previously: noise reduction is rather flat for $H > 16$ but drops off significantly for $H < 14$.

H (samples)	Comp. Time (sec.)	% Noise reduction										
		$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	Average
22	1029.5	88.77	94.01	98.16	94.30	91.72	92.17	92.38	92.63	90.79	90.01	92.49
20	1010.7	89.78	91.32	95.35	94.95	93.49	87.35	89.07	91.87	91.27	85.33	90.98
18	955.16	95.82	93.33	96.77	94.57	91.14	89.42	96.36	96.47	98.17	93.79	94.58
16	838.58	90.25	93.92	93.99	95.81	93.06	94.88	91.84	99.06	94.81	79.92	92.75
14	782.52	82.62	90.56	89.74	94.96	86.01	89.24	88.13	92.55	89.53	75.49	87.88
12	673.94	75.22	80.13	67.82	83.14	67.25	76.20	67.37	77.24	79.78	46.78	72.09
11	644.07	66.00	77.99	58.85	83.90	62.86	78.27	62.73	73.53	84.14	40.42	68.85
10	618.89	67.23	77.23	58.95	79.95	62.71	76.32	63.87	79.67	80.59	65.54	71.21
8	481.36	69.84	74.82	61.68	74.43	54.95	66.49	63.22	74.06	79.98	62.12	68.16
6	375.42	60.59	56.81	53.87	71.39	47.86	66.72	57.07	61.76	67.97	65.05	60.91
5	316.60	53.45	54.06	48.06	64.81	43.87	63.71	57.05	47.49	71.75	56.98	56.12
4	267.11	48.30	46.34	42.05	62.07	37.86	61.25	41.18	39.71	61.92	43.90	48.46

Table 4.3: Percentage of noise reduction for different values of H

H (samples)	Comp. Time (sec.)	% RMSE reduction										
		$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	Average
22	1029.5	83.71	83.93	81.10	93.53	72.60	90.60	73.61	69.87	88.95	59.05	79.70
20	1010.7	84.96	84.15	80.72	94.42	72.44	84.65	72.00	70.46	87.21	62.56	79.36
18	955.16	85.22	80.84	80.09	91.66	73.88	84.15	69.87	67.13	84.67	46.38	76.39
16	838.58	83.24	78.91	77.31	91.89	69.79	87.95	66.74	62.64	91.34	57.44	76.72
14	782.52	79.24	77.46	80.07	88.10	66.94	81.77	64.12	59.07	89.48	50.47	73.67
12	673.94	57.63	75.31	69.80	82.60	60.35	72.48	70.45	75.28	71.90	46.34	68.21
11	644.07	55.13	74.12	64.62	83.84	57.80	72.33	64.96	71.04	73.87	39.06	65.68
10	618.89	55.88	73.79	64.55	79.90	56.80	74.20	65.85	72.54	75.65	46.82	66.60
8	481.36	51.96	70.69	60.30	74.38	52.00	64.45	61.68	67.77	70.91	48.34	62.25
6	375.42	44.85	59.41	52.80	67.77	45.76	59.00	52.41	61.04	59.86	48.16	55.11
5	316.60	44.66	54.28	51.41	65.60	42.65	53.02	51.10	53.71	54.73	25.26	49.64
4	267.11	44.17	49.30	41.14	58.01	37.63	34.28	33.53	43.31	49.63	25.08	41.61

Table 4.4: Percentage of RMSE reduction for different values of H

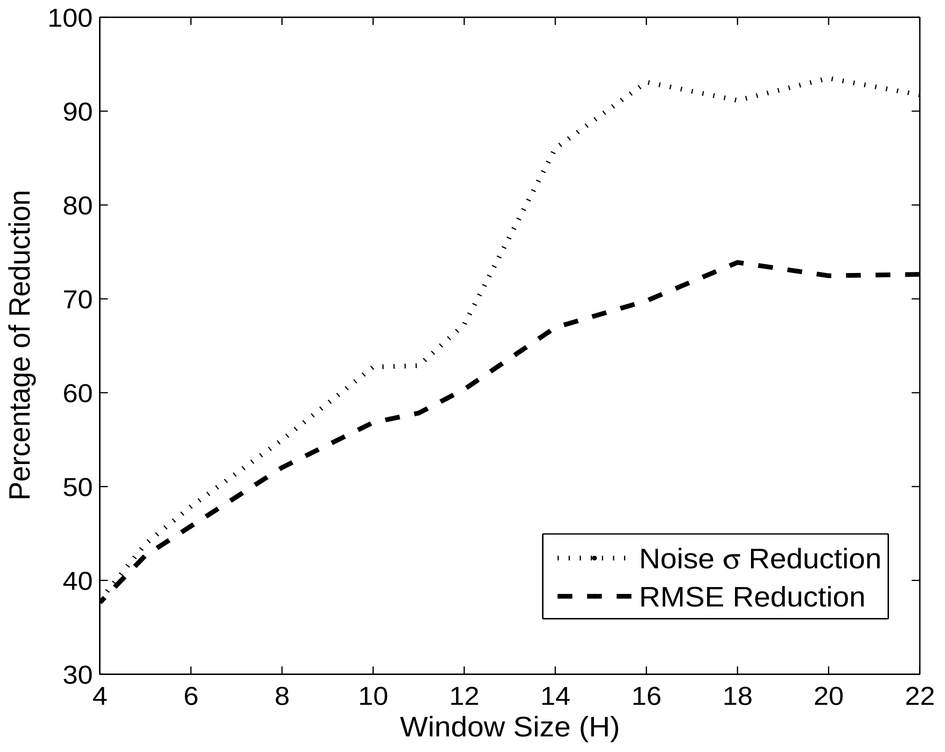


Figure 4.3: Percentage of reduction in noise and RMSE Vs H for $Fout_{treat-vap}$

In order to obtain percentages of noise reduction and RMSE reduction above 50%, the window size chosen is $H = 8$. Using this value the computation time is reduced from 68 times real time to 48. An improvement in time is achieved while obtaining a decent reduction in noise.

4.2.4 Using Scaling

The previous section demonstrated the compromise when setting the window size. Although a faster performance has been achieved, the computer time is still too large to allow the application to be implemented in real time. Thus H was set to 8 and some other experiments were done by changing other parameters to reduce time and improve performance of the NDDR.

In this section a second trial was done by changing the scaling. Initially all the values of the weights were equal and set to 1 in order to preserve the maximum-likelihood nature of the estimation scheme. However, it was considered that the optimization could be affected by the spread range of values of the variables involved in the pilot plant model. To normalize the optimization function, the value of η_i in equation (4.5) was set to the inverse of the nominal operating point value of every variable (Table 3.2).

Table 4.5 shows the outcome of the scaling test. The results show that using scaling provides a faster and in general more efficient reconciliation. The final time for the algorithm is $t = 10.05$ *sec.*, and the computation time was $t_{comp} = 425.4$ *sec.*, which is lower than in the previous test performed using $H = 8$ and without scaling ($t_{comp} = 481.36$ *sec.*). In general the majority of the variables exhibit a greater reduction in noise and RMSE as well.

	Variable	NO Scaling		Scaling	
		% σ Reduction	% RMSE Reduction	% σ Reduction	% RMSE Reduction
Inputs	$Fout_{sep-liq}$	69.84	51.96	81.08	74.08
	$Fout_{sep-vap}$	74.82	70.69	67.08	63.61
	$Fout_{treat-wat}$	61.68	60.30	88.03	75.71
	$Fout_{treat-oil}$	74.43	74.38	79.72	77.66
	$Fout_{treat-vap}$	54.95	52.00	58.97	59.88
Outputs	$V_{sep-liq}$	66.49	64.45	73.68	72.42
	$P_{sep-vap}$	63.22	61.68	63.42	56.35
	$V_{treat-wat}$	74.06	67.77	51.91	44.50
	$V_{treat-oil}$	79.98	70.91	81.72	75.40
	$P_{treat-vap}$	62.12	48.34	66.73	31.21
Average		68.16	62.25	71.23	63.08

Table 4.5: NDDR results using scaling

4.2.5 Modifying Initial Guesses

The initial guess for the optimal solution to the NLP problem was another factor modified in order to obtain a faster optimization. According to previous studies ([16] and [1]), having good initial guesses can improve the robustness of the algorithm. The estimates from the previous time step are the most reasonable initial guesses, and they were used for all the previous experiments. In order to assess the impact of changing initial guesses, several possibilities were evaluated. These included setting initial guesses to the mean of the measurements in the previous window, the previous measurements and the estimates at the beginning of the window.

The results are shown in tables 4.6 and 4.7, the running time for this test is $t_f =$ 10.05 *sec.* It can be observed that the best performance with the fastest results are obtained with the previous estimates as initial guesses, as expected.

Initial Guesses	Comp. Time (sec.)	%Noise reduction										
		$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	Average
Previous estimate	481.36	69.84	74.82	64.68	74.43	54.95	66.49	63.22	74.06	79.98	62.12	68.16
Mean previous window	529.65	79.41	73.83	89.14	87.65	62.93	98.75	91.05	94.85	97.55	41.67	81.68
Previous measurement	815.47	37.59	14.34	66.30	88.51	54.49	90.67	58.57	90.94	86.64	87.23	67.53
Estimate at start of window	770.20	48.17	79.54	75.49	79.17	51.45	96.15	93.35	94.11	94.45	93.08	80.49

Table 4.6: % Noise reduction using different initial guesses

Initial Guesses	Comp. Time (sec.)	%Noise reduction										
		$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	Average
Previous estimate	481.36	51.96	70.69	60.30	74.38	52.00	64.45	61.68	67.77	79.91	48.34	62.25
Mean previous window	529.65	73.30	68.14	77.26	79.08	63.32	98.02	87.13	76.05	70.31	52.49	74.51
Previous measurement	815.47	1.74	7.77	35.49	32.93	57.74	93.64	67.45	81.99	88.40	-82.40	38.48
Estimate at start of window	770.20	39.93	70.24	75.07	81.14	59.76	95.34	82.59	47.40	82.46	-0.35	63.36

Table 4.7: % RMSE reduction using different initial guesses

4.2.6 Change in Optimization Algorithm's Parameters

The next step in trying to find a better performance for the NDDR was to modify some parameters that govern the behavior of `fminunc`. These changes included modifying the termination tolerance on the objective function value (default $1e-3$) and the termination tolerance on the estimates, $\hat{y}$ (default $1e-3$). Several tests were performed in order to establish the effect of these parameters on the performance of the algorithm. Table 4.8 shows the results for some of the tests executed.

Optimization Parameters	Comp. Time (sec.)	% of Reduct.	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	Average
T $\hat{y} = 1e-3$	481.37	Noise	69.84	74.82	61.68	74.43	54.95	66.49	63.22	74.06	79.98	62.13	68.16
T Fun $= 1e-3$		RMSE	51.96	70.71	60.35	74.38	52.01	64.45	61.69	67.77	70.92	48.34	62.25
T $\hat{y} = 1e-2$	360.11	Noise	71.18	72.37	59.73	75.77	57.21	66.34	54.90	66.12	81.94	36.03	64.15
T Fun $= 1e-2$		RMSE	50.58	68.26	58.56	74.23	55.32	63.19	53.80	65.69	73.62	11.55	57.48
T $\hat{y} = 1e-1$	283.11	Noise	67.10	73.68	53.38	75.39	51.21	64.30	60.17	61.40	79.31	22.65	60.85
T Fun $= 1e-1$		RMSE	48.69	77.52	47.49	79.20	34.39	55.61	62.31	73.98	75.12	-87.7	46.65
T $\hat{y} = 1$	700.37	Noise	99.80	99.85	99.80	93.07	99.80	90.89	41.33	67.95	74.35	10.98	77.78
T Fun $= 1$		RMSE	-72.8	83.09	91.26	30.31	-224	87.11	-57.4	67.68	-0.15	-3566	-356
T $\hat{y} = 1e-4$	818.67	Noise	69.07	73.36	62.55	75.57	57.52	71.08	56.92	70.83	76.30	55.18	66.83
T Fun $= 1e-4$		RMSE	53.34	70.88	61.88	75.33	53.33	68.40	58.42	67.05	69.63	52.64	63.09

Table 4.8: NDDR results changing optimization parameters

It was observed that for larger values of tolerance, the computation time was shorter, as expected. However the level of bias, reflected in the RMSE, increases. The percentage of noise reduction for tolerances of $1e-3$, $1e-2$, and $1e-1$ is similar for all the variables except for $P_{treat-vap}$, which reconciliation is negatively affected when the tolerance is increased. For larger values of tolerance, the estimates of $P_{treat-vap}$ present more noise and more bias level. When the tolerance is equal to 1 the noise reduction seems to be optimal for the majority of the variables, reaching values close to 100%; nevertheless, the reconciliation is not correct because a high level of bias is added to the estimates.

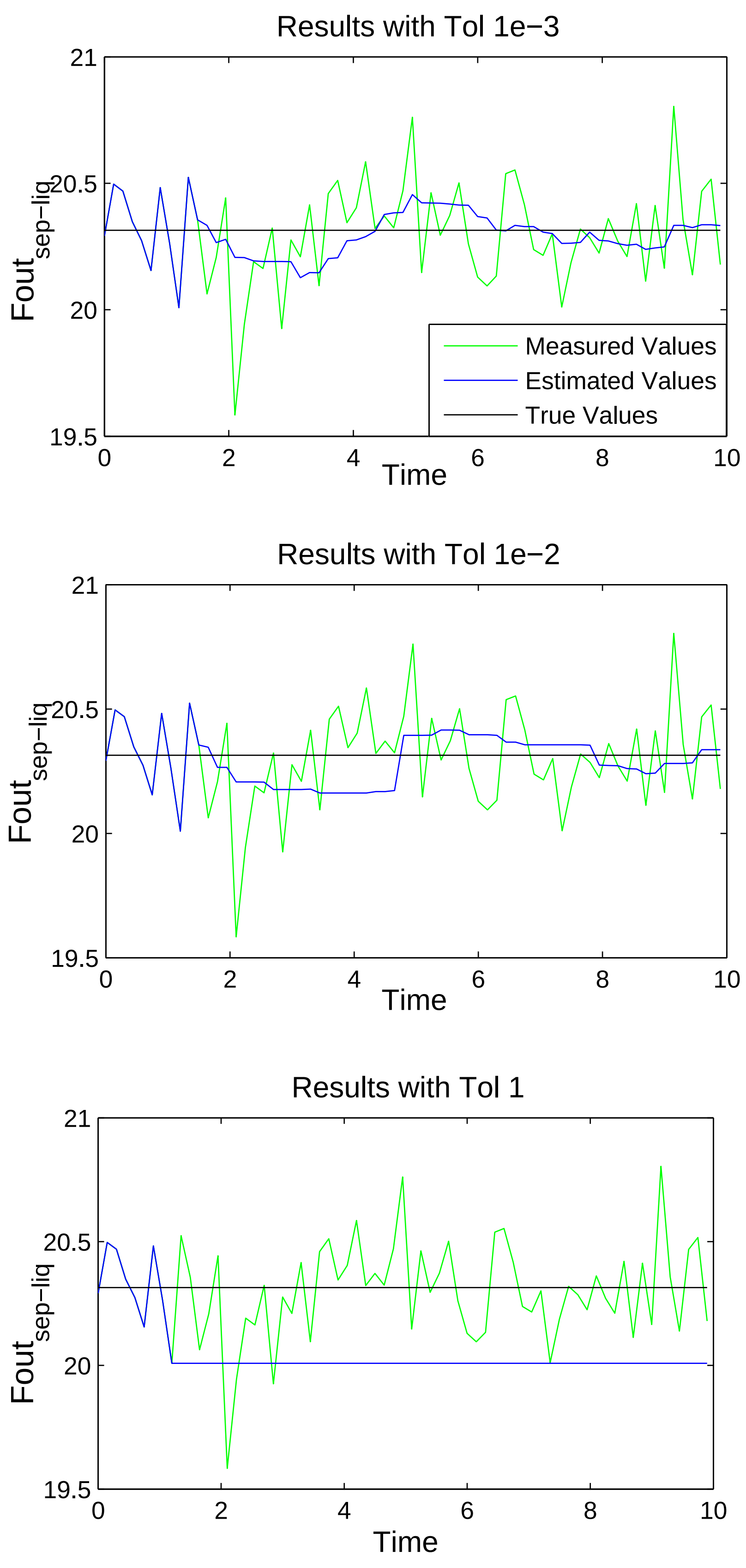


Figure 4.4: Results for $Fout_{sep-liq}$ with difference optimization parameters

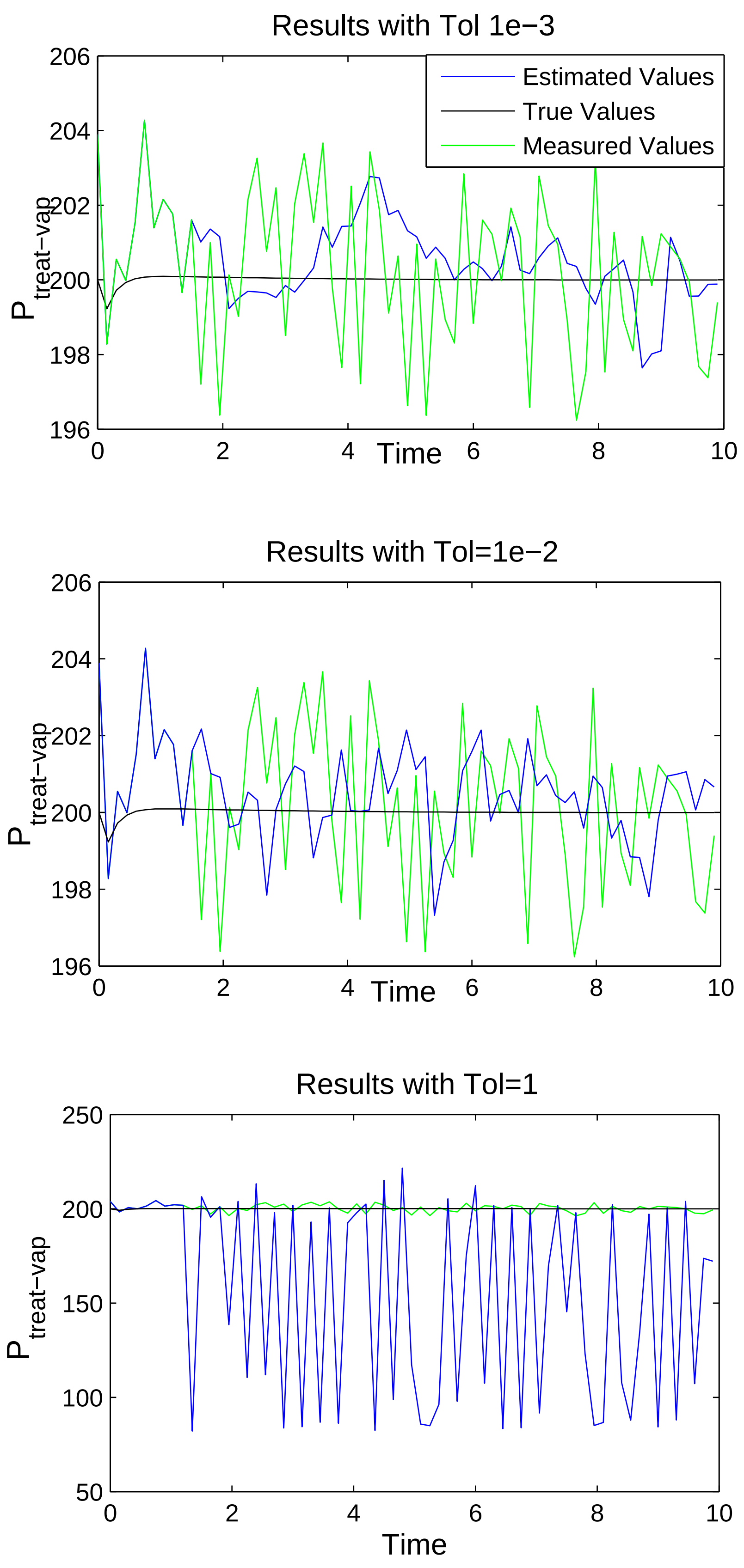


Figure 4.5: Results for $P_{treat-vap}$ with difference optimization parameters

Figures 4.4 and 4.5 show the response of the variables $Fout_{sep-liq}$ and $P_{treat-vap}$ to the reconciliation algorithm using values of $1e-3$, $1e-2$, and 1, respectively. It can be observed the increment in the bias level in $Fout_{sep-liq}$ and the negative effect on noise and RMSE reduction for $P_{treat-vap}$. In the case of the variable $Fout_{sep-liq}$ with a tolerance of 1, the estimate is completely flat; this is because the tolerance is large enough to be satisfied with the first initial guess, so the algorithm does not need to find a new value. When the tolerance of the optimization algorithm is increased, the minimization process is truncated, producing inaccurate estimates. The effect of this phenomenon is more evident in variables with fast dynamics such as, in our particular case, $P_{treat-vap}$.

4.2.7 NDDR Results in Scenarios with Set Point Changes

The best compromise in performance was obtained for the size of window (H) equal to eight, using the previous estimates as initial guesses, applying scaling in the optimization routine, and using the default values for the optimization algorithm parameters. These conditions are selected as part of the definitive NDDR algorithm. In order to show the effectiveness of this NDDR algorithm two different scenarios are presented:

In the first scenario, the simulation runs for a time $t_f = 100$ *sec.* The plant starts working at the nominal operating point, and a positive setpoint change of 2% is applied at the time $t = 50$ *sec.* Table 4.9 and figure 4.6 present the results obtained. The noise added to the simulation values has a standard deviation of 1% nominal operating value of each variable. This study required a computation time of $t_{comp} =$ 4827.87 *sec.* or 48.27 times real time to complete the reconciliation.

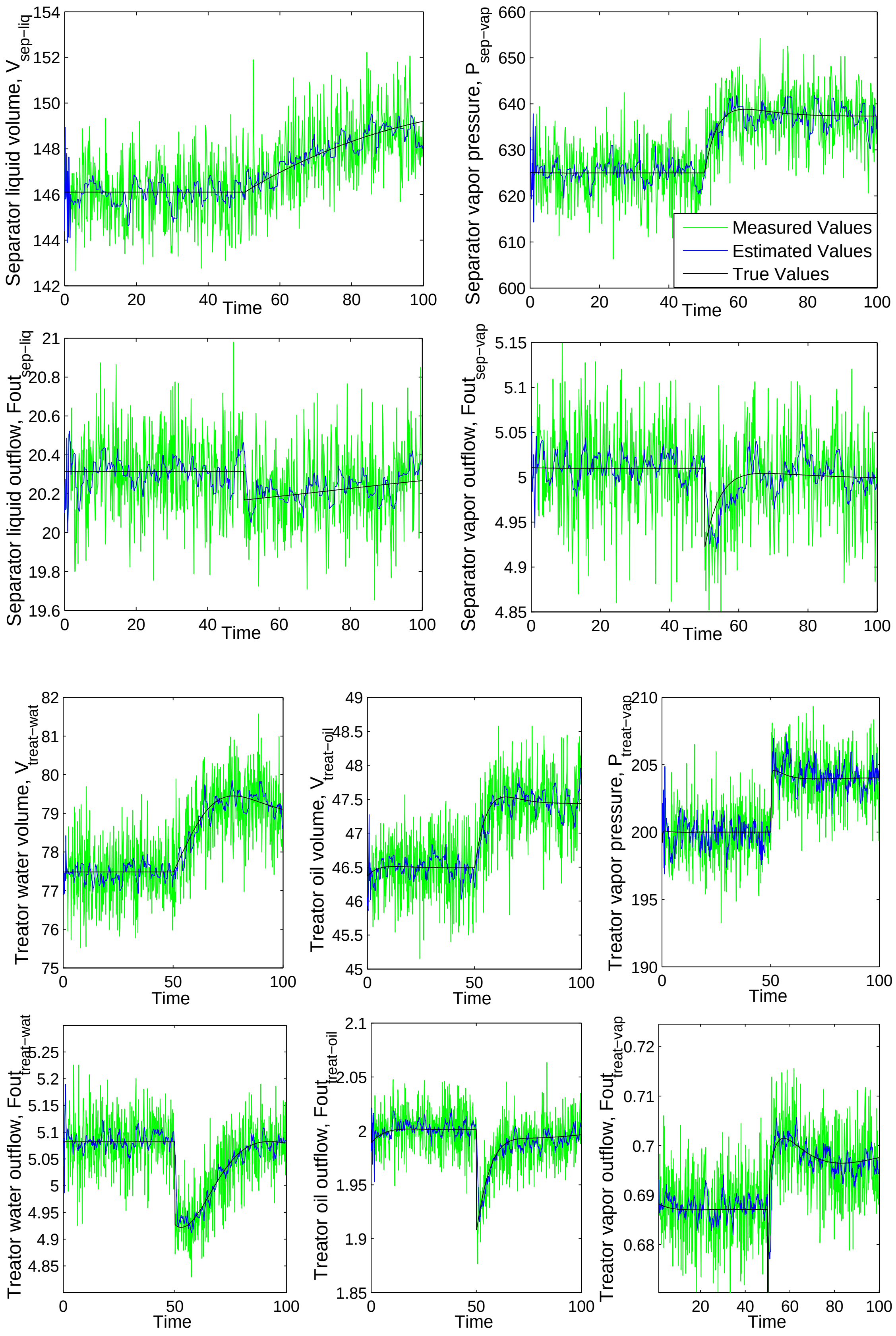


Figure 4.6: NDDR results for a positive set point change

Percentage of reduction	Variables										
	Inputs					Outputs					Average
	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	
Noise	73.52	75.57	73.13	70.49	68.52	63.20	68.47	78.64	77.88	56.51	70.59
RMSE	73.30	75.06	73.85	64.89	70.10	63.23	65.25	72.80	77.52	52.94	68.89

Table 4.9: NDDR results for a positive set point change

The second scenario illustrates the application of the NDDR algorithm for a case when a larger negative setpoint change of 5% is applied at the time $t = 50$ *sec.*, in a run with $t_f = 90$ *sec.* The results are presented in Figure 4.7 and table 4.10. The computation time for this run is $t_{comp} = 4261.76$ *sec.*; this is equivalent to 47.32 times the real time.

Percentage of reduction	Variables										
	Inputs					Outputs					Average
	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	
Noise	64.76	65.57	65.46	65.77	68.53	69.23	61.75	69.11	69.13	60.01	65.93
RMSE	65.11	64.64	55.75	44.15	26.61	68.89	61.50	68.49	68.32	53.12	57.66

Table 4.10: NDDR results for a negative set point change

It can be observed that for both cases the estimate values contained far less noise than the simulated measurements. The NDDR algorithm is able to reduce noise and reduce RMSE in transient and steady state conditions.

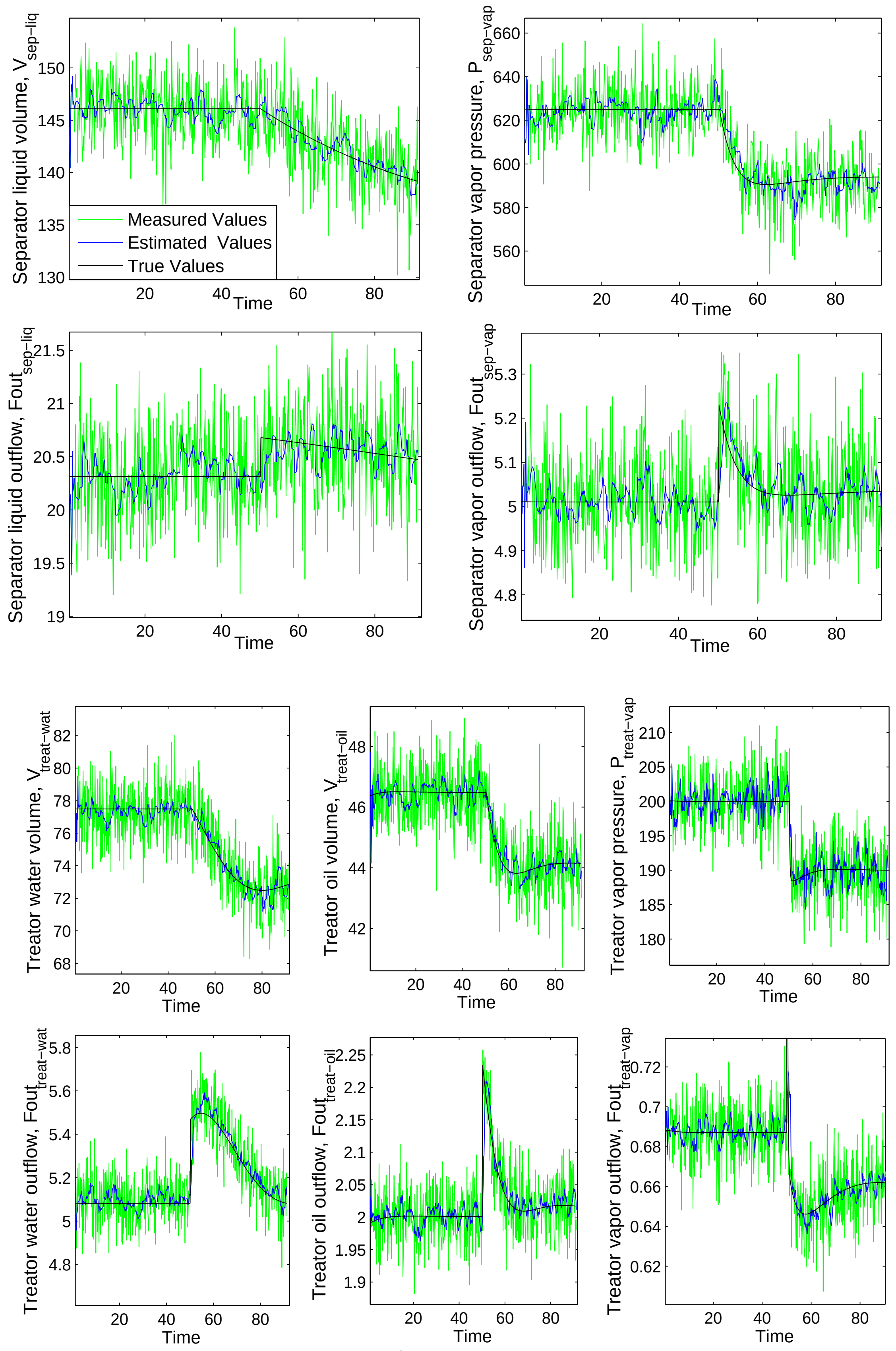


Figure 4.7: NDDR results for a negative set point change

4.3 NDDR vs Low Pass Filtering

The main difference between data reconciliation and other filtering methods is that DR uses the model constraints to obtain the estimates of the process variables. The measurements are adjusted so that the estimates meet the constraints. In that way, the estimates present less noise than the measurements and at the same time they fulfill the relationship between process variables established by the model constraints, including material and energy balance. Even though standard filtering techniques reduce noise significantly, they may introduce some dynamic lag in the response or they may reduce overshoot or undershoot after a step change, especially in process variables with a fast dynamic behavior. The following examples show a comparison between the performance of the NDDR algorithm and a low pass filter.

The filter used was an exponential filter, which is one of the simplest forms of linear recursive filter. The exponential filter is described by the following expression:

$$z_k = \alpha z_{k-1} + (1-\alpha) y_k \tag{4.9}$$

where

- z_k is the output of the filter at the current time;
- z_{k-1} is the output of the filter at the previous time step;
- y_k is the input of the filter;
- $0 \leq \alpha \leq 1.0$ is the parameter of the filter.

Equation 4.9 indicates that the filtered measurement is a weighted sum of the current measurement z_k and the filtered value at the previous sampling instant z_{k-1}. The filter constant α is defined as:

$$\alpha \doteq \frac{\Delta t}{\tau + \Delta t} \tag{4.10}$$

where τ is the time constant of each variable, which are listed in table 3.1, and Δt is the sampling time, which is set to $\Delta t = 0.15$ *sec*. The value of the filter constant dictates how strong the filtering action will be. If $\alpha \rightarrow 1$, the degree of filtering is larger and the measurement does not play a big role in the calculation of the average. On the other extreme, if $\alpha \rightarrow 0$ less filtering is being performed. The filter constants calculated according with these values are presented in table 4.11.

Filter Constants	Variables									
	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$
Calculated	0.9958	0.9543	0.9856	0.9592	0.4692	0.9958	0.9543	0.9856	0.9592	0.4692

Table 4.11: Filter constants - Filter One

The filter was applied to the pilot plant model and the filtered signal was compared with the estimated values obtained with NDDR. For this case, the simulation was initialized with all the variables at the nominal operating point, and at time $t = 50$ *sec.* a positive setpoint change of 2% was applied. Figure 4.8 show the performance of the low pass filter using the constants presented in table 4.11. Table 4.12 shows the comparison between the performance of the filter and the performance with the NDDR. Comparing the filtered signals with the estimated values it is observed that although the filter is more efficient at reducing the noise in the signals, there is a large dynamic lag introduced in the filtered values. However, this is not a reasonable comparison, given that the filtered signal contain less noise than the estimated values.

In order to have a fair comparison, the filter constants were adjusted to obtain approximately as much noise reduction with the filter as it was achieved with the NDDR algorithm. The filter constants used are presented in table 4.13, and the noise reduction obtained is presented in table 4.14. The previous scenario was executed again with the new filter parameters, and the outcomes from the filter and from the NDDR

Percentage of Noise reduction	Variables										
	Inputs					Outputs					Average
	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	
Filter	78.11	86.54	82.64	87.90	79.57	78.26	88.25	93.33	73.17	79.00	82.68
NDDR	63.49	61.87	62.85	450.61	68.52	63.83	68.47	68.02	65.48	56.51	62.97

Table 4.12: Noise reduction comparison between filter one and NDDR

were compared. It can be observed that the filter introduced a smaller dynamic lag compared with the previous scenario but still is bigger than the one obtained with NDDR. A closer view of the variables $Fout_{treat-wat}$, and $Fout_{treat-oil}$ is shown in appendix D, in figures D.1 and D.2 respectively. Table 4.14 shows the quantitative comparison.

Filter Constants	Variables									
	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$
Adjusted	0.82	0.9	0.9	0.89	0.8	0.8	0.85	0.75	0.8	0.85

Table 4.13: Filter constants - Filter two

Percentage of Noise reduction	Variables										
	Inputs					Outputs					Average
	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	
Filter	65.44	70.34	63.82	52.52	41.71	67.50	69.89	63.43	66.17	60.47	62.14
NDDR	63.49	61.87	62.85	450.61	68.52	63.83	68.47	68.02	65.48	56.51	62.97

Table 4.14: Noise reduction comparison between filter two and NDDR

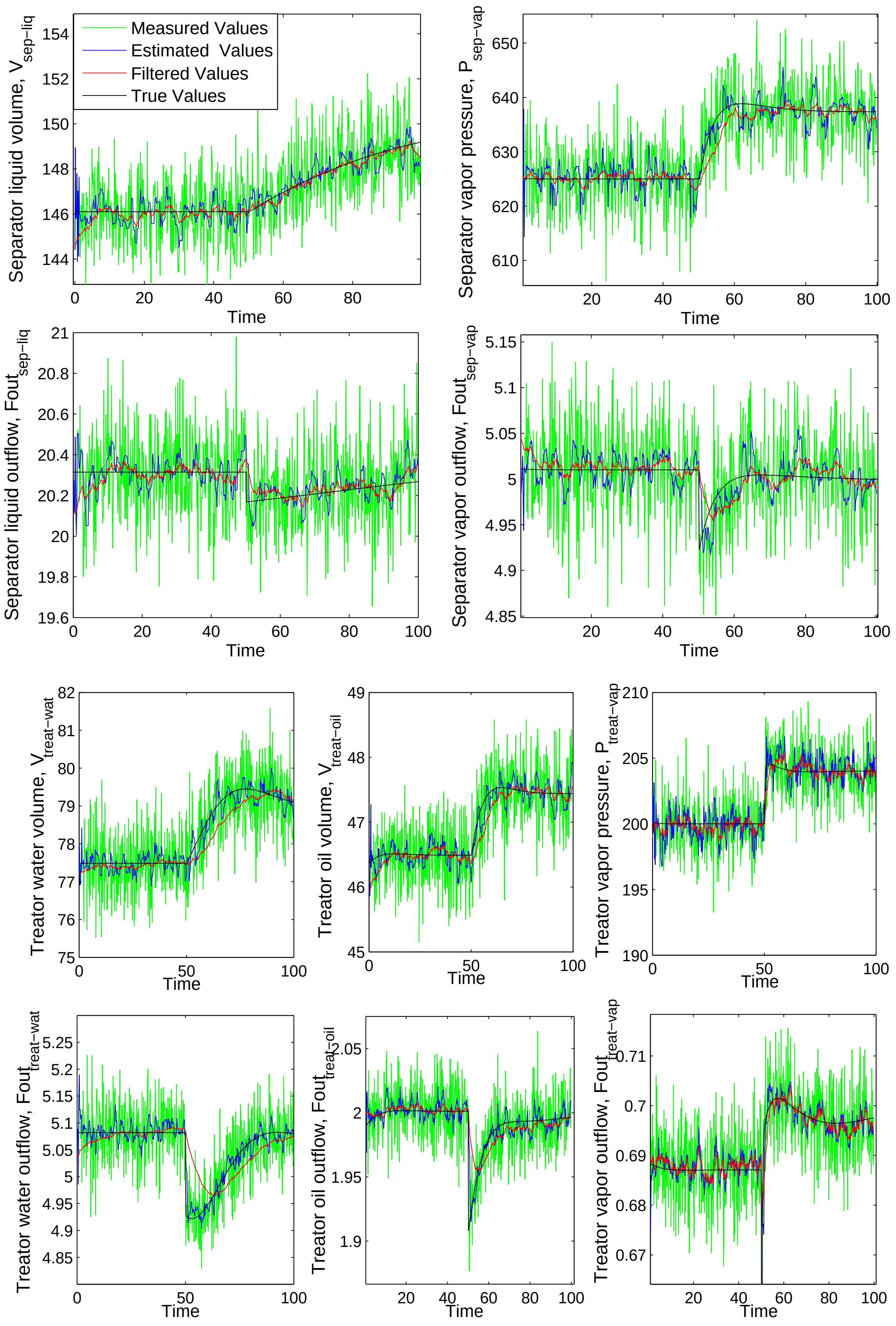


Figure 4.8: Separator signals filtered with low-pass filter

Chapter 5

A Hybrid Approach to Solving Dynamic Data Reconciliation

The results presented in chapter 4 show that the estimates obtained with NDDR are significantly smoother than the corresponding measurement. However, the time consumed by the algorithm is still too large. Another drawback to this approach is the need for a nonlinear process dynamic model, which is often not available or not practical to develop. In order to reduce the computation time and address the model availability issue, a new hybrid approach was developed and tested. The main idea is to use two different methods to perform the data reconciliation according to the state in which the system is: transient or steady state.

Using the Steady-state algorithm explained in chapter 3, it is possible to know the state of the system. During transient periods, the data reconciliation is executed using the linearized model of the plant provided by the System Identification Agent [38] operating on perturbation variables, *i.e.*, $\delta y = y - y_0$ where y_0 is the steady state operating point. This approach is valid for small changes in the set-points and maintains the goal of DR, namely energy and material balance is retained in y_0. In the steady state intervals, the data reconciliation is performed using an equilibrium finding approach. Instead of solving numerically the differential equations that describe the system over the window, an equilibrium finding algorithm is used to determine the value of the states at the equilibrium point. Subsequently, the value of the outputs for the equilibrium point are calculated. It is faster to obtain the equilibrium point than to run the model for ten samples every iteration of the optimization loop. However, using a dynamic model for equilibrium finding is still substantially less efficient than using a static process model directly. Thus the real promise of the hybrid approach will be realized in applications for which a static model exists, which is not true for the pilot plant. The following is a feasibility study, to demonstrate that the hybrid approach works.

The nonlinear pilot plant model is defined in terms of the differential equation as $\dot{x} = f(x, u)$. For an input value u_0 and its corresponding equilibrium x_0, the output value is $y_0 = h(x_0, u_0)$. The perturbation variables can be defined as $\delta x = x - x_0$, $\delta u = u - u_0$, and $\delta y = y - y_0$. If the perturbations are small and if continuous partial derivatives exist at (x_0, u_0), the behavior of the original nonlinear system near x_0 is similar to that of:

$$\delta \dot{x} = A\delta x + B\delta u \tag{5.1}$$

$$\delta y = C\delta x + D\delta u \tag{5.2}$$

where $A = [\frac{\delta f}{\delta x}]_{x_0,u_0}$, $B = [\frac{\delta f}{\delta u}]_{x_0,u_0}$, $C = [\frac{\delta h}{\delta x}]_{x_0,u_0}$, and $D = [\frac{\delta h}{\delta u}]_{x_0,u_0}$ [43]. A, B, C and D are obtained by the System Identification Agent [38].

In transient state, for small changes in the setpoint, the linearized model can be used as the constraints in equation (4.2). However, in steady state it is not accurate to use a linear approximation, given that the equilibrium point for the linearized model is different than the equilibrium point for the nonlinear model. Furthermore, the concept of data reconciliation is to adjust the measurements according to conservation laws constraints. This definition would not be satisfied by using the linearized model, because although the behavior of the original nonlinear system is similar to the linear model the linear model does not provide the physical relationship between inputs and outputs. Therefore the conservation laws constraints would not be fulfilled.

For the steady state case, the plant has reached an equilibrium. This is why it is valid to calculate the estimates as the response of the plant to the equilibrium point value. Once the equilibrium state values (x_0) are calculated using an equilibrium finding algorithm, it is possible to obtain the value of the outputs (y_0). This procedure is faster than to running ten-point simulations of the plant in every loop of the NDDR optimization.

5.1 Hybrid Approach Results

The scenario used to test the hybrid approach starts at the nominal operating point and at the time $t = 50$ *sec.* a positive set point change of 2% is applied to all the variables. The sampling time is $\Delta t = 0.15$ *sec.* and the final time is $t_f = 300$ *sec.* Figure 5.1 show the results for the test. The computation time is $t_{comp} = 7494.30$ *sec.*; this correspond to twenty four times the final time.

It can be observed that the estimates present less noise than the measurements, in both the transient and the steady state. In the moment of the transition from transient to steady state ($t = 249.9$ *sec.*), outliers are present in some of the variables. It takes a window of H samples for the system to adapt again using the nonlinear dynamic model. Table 5.1 show the quantitative results for this test, confirming the results mentioned above. The fastest variables ($P_{sep-vap}$ and $P_{treat-vap}$) are the most affected by the the transition.

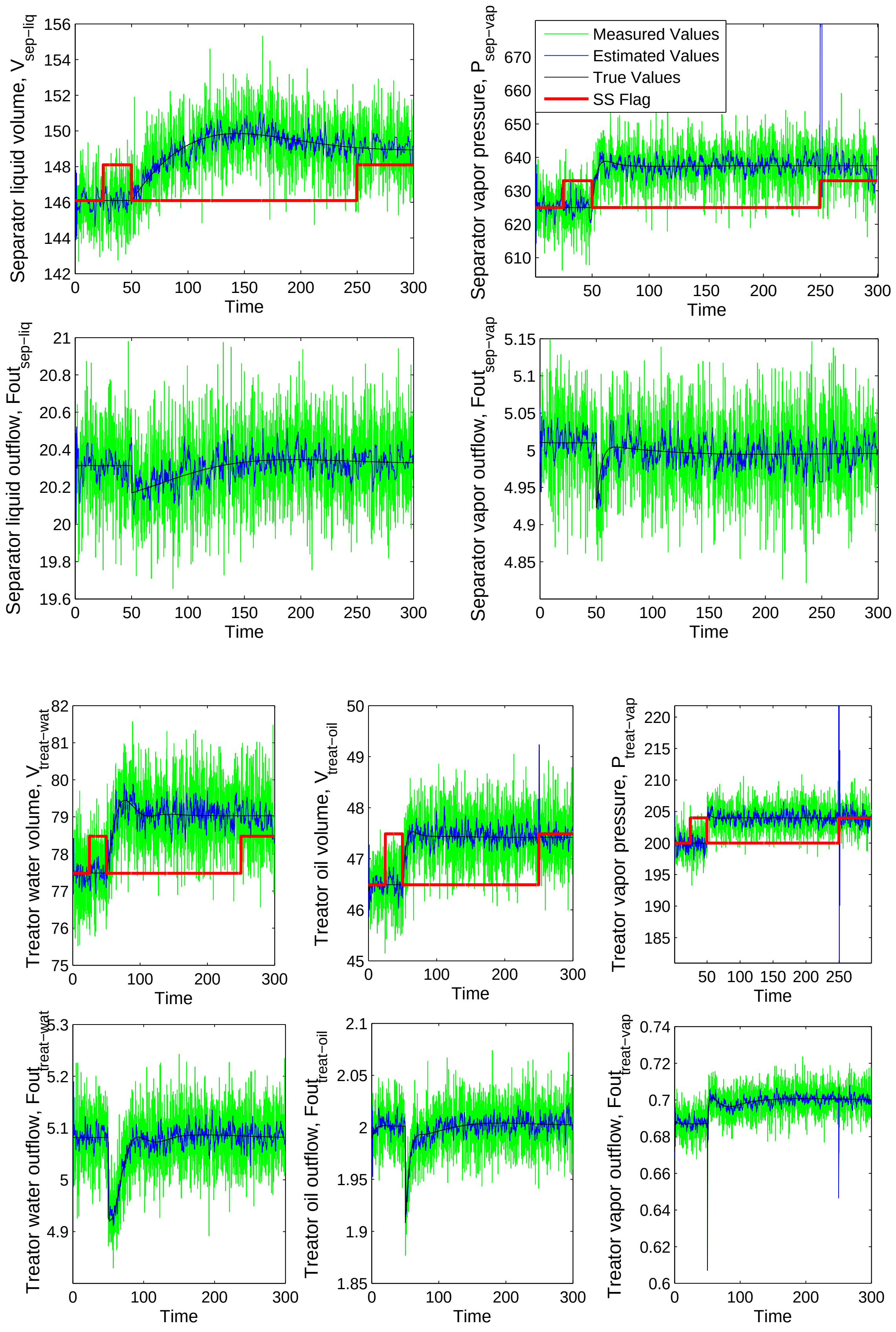


Figure 5.1: Hybrid NDDR results

In this test the computation time is shorter than the time consumed in the examples presented in chapter 4 (approximately 49 times real time). However, this is due to the long transient period, where the linearized model is used. Using the equilibrium approach with the current pilot plant model derived by Sayda and Taylor [2] does not substantially improve the time consumed for the data reconciliation algorithm. Nevertheless, if a simpler nonlinear steady state model based on material energy balance would be available, the equilibrium algorithm would be more efficient and it could be used in the hybrid approach to obtain a faster and reliable data reconciliation routine. The derivation of the new nonlinear steady state model is not part of the scope of this project and it is suggested as future work.

Percentage of reduction	Final Time (sec.)	Variables										Average
		Inputs					Outputs					
		$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	
Noise	7494.3	70.21	67.14	68.90	63.01	50.83	68.74	-23.53	67.97	63.59	-103.08	39.38
RMSE	7494.3	69.94	67.05	68.94	63.07	50.90	68.73	-23.46	68.04	63.30	-102.73	39.37

Table 5.1: Hybrid approach results

5.2 Solving the Steady-state/Transient Transition Problem

A routine, which detects changes in the steady state flag and analyzes the values of the estimates during the succeeding window, was designed. The objective of this routine is to solve the problem with the transition from transient state to steady state, eliminating the outliers produced by the switching between the models. When the steady state flag changes from unity to zero or *vice versa*, the algorithm starts to compare the current estimate (z_c) with the previous one (z_{c-1}). If the difference between them is larger than five times the standard deviation of the previous data window, then the z_c is replaced by z_{c-1}. Table 5.2 shows the noise and RMSE reduction for this new approach, and figure 5.2 shows the results for $P_{treat-vap}$ using the hybrid approach and the routine to eliminate the outliers. It can be observed that the transition is conducted without the negative effects in the estimates that are apparent in figure 5.1.

Percentage of reduction	Final Time (sec.)	Variables										Average
		Inputs					Outputs					
		$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$	$V_{sep-liq}$	$P_{sep-vap}$	$V_{treat-wat}$	$V_{treat-oil}$	$P_{treat-vap}$	
Noise	7490.1	69.24	68.36	70.34	68.65	69.36	68.20	68.34	67.79	67.85	63.31	68.15
RMSE	7490.1	68.93	67.38	67.68	61.31	52.42	68.42	67.94	68.00	67.80	60.75	65.06

Table 5.2: Hybrid approach results after solving transition problem

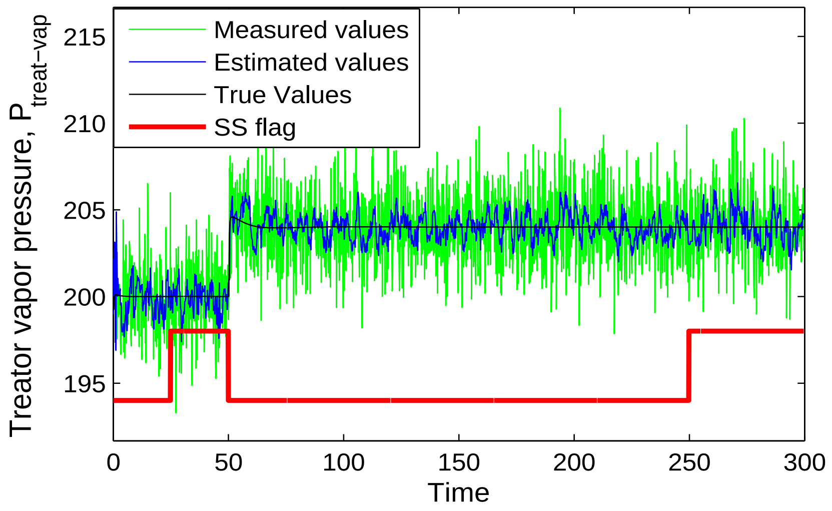


Figure 5.2: Hybrid NDDR results solving transition problem for $P_{treat-vap}$

Chapter 6

Gross Error Detection and Removal (GEDR) and Adaptive NDDR

The technique presented in chapter 4 assumed that only random errors are present in the data. However, this assumption is not valid given that gross errors may occur, caused by non-random events such as instrument malfunctioning, miscalibration, corrosion in sensors, faulty analog-digital conversion, etc. Even though gross errors are less common, if not removed, they affect the complete reconciliation. The objective is not only to identify the presence of gross error but also correct it.

Laylabadi and Taylor [1], proposed the adaptive NDDR and Novel GEDR techniques in order to give solution to two different but related problems. The first is to apply NDDR to systems with unknown error statistical models and the second is to detect and to remove the gross errors. These techniques are explained in this chapter and the results of implementing the combined ANDDR + GEDR algorithm in the pilot plant model are presented. The contribution in this area is extending and/or refining the methods in [1] to make them applicable in more realistic industrial conditions.

6.1 Gross Error Detection and Removal

The method used to solve GEDR utilizes the same moving-horizon window approach used for the basic NDDR. The technique is based on calculating the difference $d_{c,i}$ between each new measurement $\tilde{y}_{c,i}$ and the previous value of the mean. Assuming a statistical model is available, the difference is compared at each time step with a multiple of the previous standard deviation, $\sigma_{c-1,i}$. If $|d_{c,i}|$ is bigger than the threshold, then the system detects the presence of a gross error and proceeds to replace it with the previous estimate. The subscript c corresponds to the number of the sample and the subscript i represent the number of the variable.

6.2 Adaptive Nonlinear Dynamic Data Reconciliation ANDDR

Most DR methods assume gross-error free measurements and known statistical models, but this is idealistic and difficult to find in real plants. The Adaptative NDDR + GED was designed to work in cases where there is not a statistical model for the noise and with data which may contain outliers. The method consists of applying first the GEDR technique, explained above, to eliminate the undesired outliers and subsequently to calculate a new standard deviation based on the new measurements. In this case, measurements in the window are used to calculate the standard deviation $\sigma_{c,i}$ and the mean value $\hat{m}_{c,i}$. The method is based on an assumption of isolated gross errors, this means gross errors do not happen in consecutive samples. The following equations summarize the procedure:

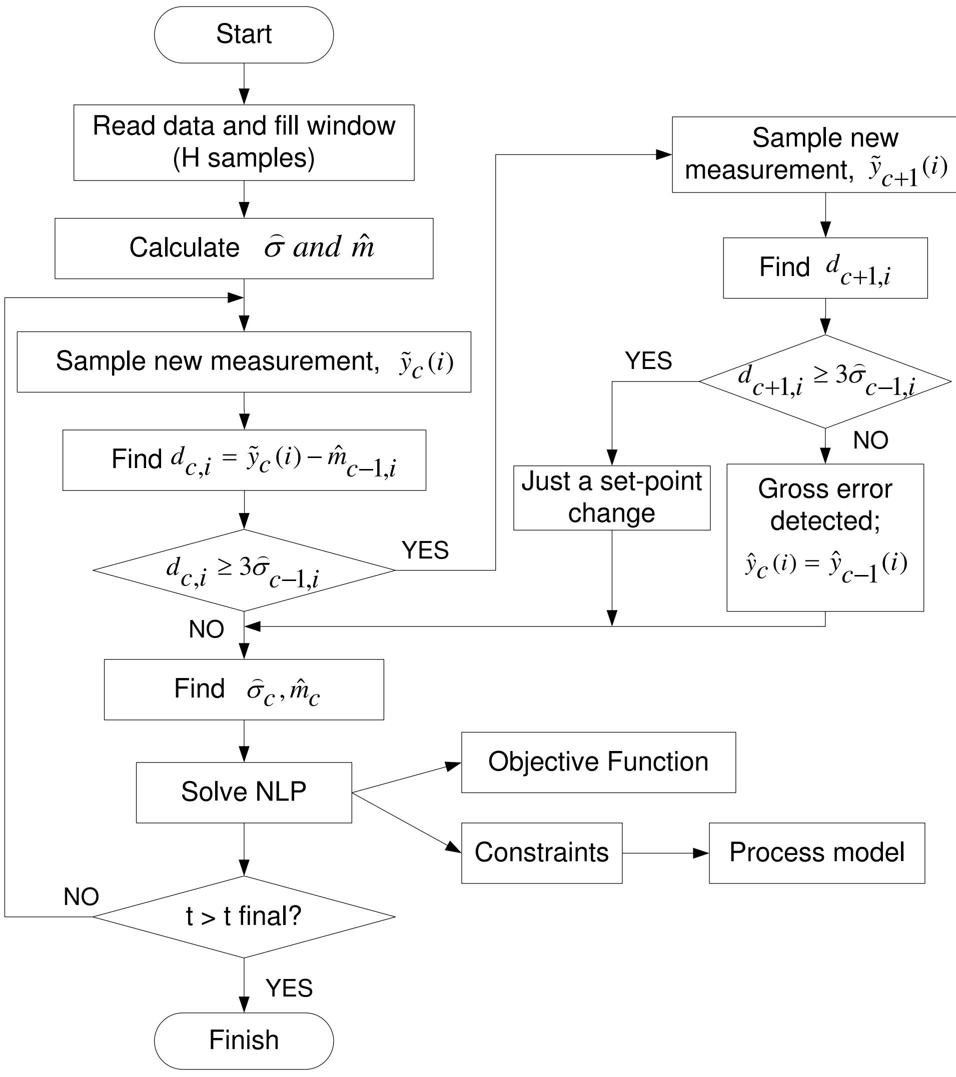
$$d_{c,i} = \tilde{y}_{c,i} - \hat{m}_{c-1,i} \tag{6.1}$$

$$\text{If } |d_{c,i}| > 3\sigma_{c-i,i} \text{ then } \tilde{y}_{c,i} \text{ is a Gross Error} \tag{6.2}$$

$$\hat{m}_{c,i} = \sum_{j=c-H}^{c} \left(\frac{\tilde{y}_{i,j}}{H+1}\right) \tag{6.3}$$

$$\hat{\sigma}_{c,i} = \sqrt{\sum_{j=c-H}^{c} \left(\frac{(\tilde{y}_{i,j} - \hat{m}_{c,i})^2}{H+1}\right)} \tag{6.4}$$

The flowchart presented in Figure 6.1 shows a review of the complete method consisting of ANDDR and GEDR.



Start
Read data and fill window (H samples)
Calculate $\hat{\sigma}$ and $\hat{m}$
Sample new measurement, $\tilde{y}_c(i)$
Find $d_{c,i} = \tilde{y}_c(i) - \hat{m}_{c-1,i}$
$d_{c,i} \geq 3\hat{\sigma}_{c-1,i}$
YES
NO
Sample new measurement, $\tilde{y}_{c+1}(i)$.
Find $d_{c+1,i}$
$d_{c+1,i} \geq 3\hat{\sigma}_{c-1,i}$
YES
NO
Just a set-point change
Gross error detected; $\hat{y}_c(i) = \hat{y}_{c-1}(i)$
Find $\hat{\sigma}_c, \hat{m}_c$
Solve NLP
Objective Function
Constraints
Process model
t > t final?
NO
YES
Finish

Figure 6.1: ANDDR and GEDR flowchart [1]

6.3 ANDDR + GEDR Results

This section shows the performance of the ANDDR + GEDR algorithm. The measurements were obtained by simulation of the pilot plant and addition of gaussian noise and non-random gross errors. The plant was ran for a time $t_f = 80$ *sec.*; initially the plant starts at the nominal operating point and at time $t = 50$ *sec.* a positive set point change of 2% is applied. Gaussian noise with zero mean and a standard deviation of 1% is added to the simulated data. Different isolated non random errors were added to every variable.

Figure 6.2 shows the results obtained when gross error are present in measurements. The black line shows the true values, the green line represents the noisy measurements, the red line corresponds to the estimates obtained when the ANDDR + GEDR algorithm is used to implement the data reconciliation, and the blue line represents the estimates obtained by applying the basic NDDR algorithm without gross error detection. The interval $40 \leq t \leq 70$ *sec.* is plotted, to more clearly show the benefit of GEDR. As expected, both estimates contained far less noise than the simulated measurements. In the case of ANDDR + GEDR estimates, the gross errors were successfully detected and removed. However, it can be observed that there is a significant effect of the gross errors in the basic NDDR results. The outliers produce substantial deviations in the basic NDDR estimates for H points after the event. For example, the input variable $Fout_{sep-liq}$ has gross errors at times $t = 46.35$ *sec.*, $t = 52.2$ *sec.*, $t = 57.9$ *sec.*, and $t = 66.6$ *sec.* in the time span plotted; it can be observed that at these times, the NDDR estimates (blue lines) are perturbed as long as the data window contains the gross error.

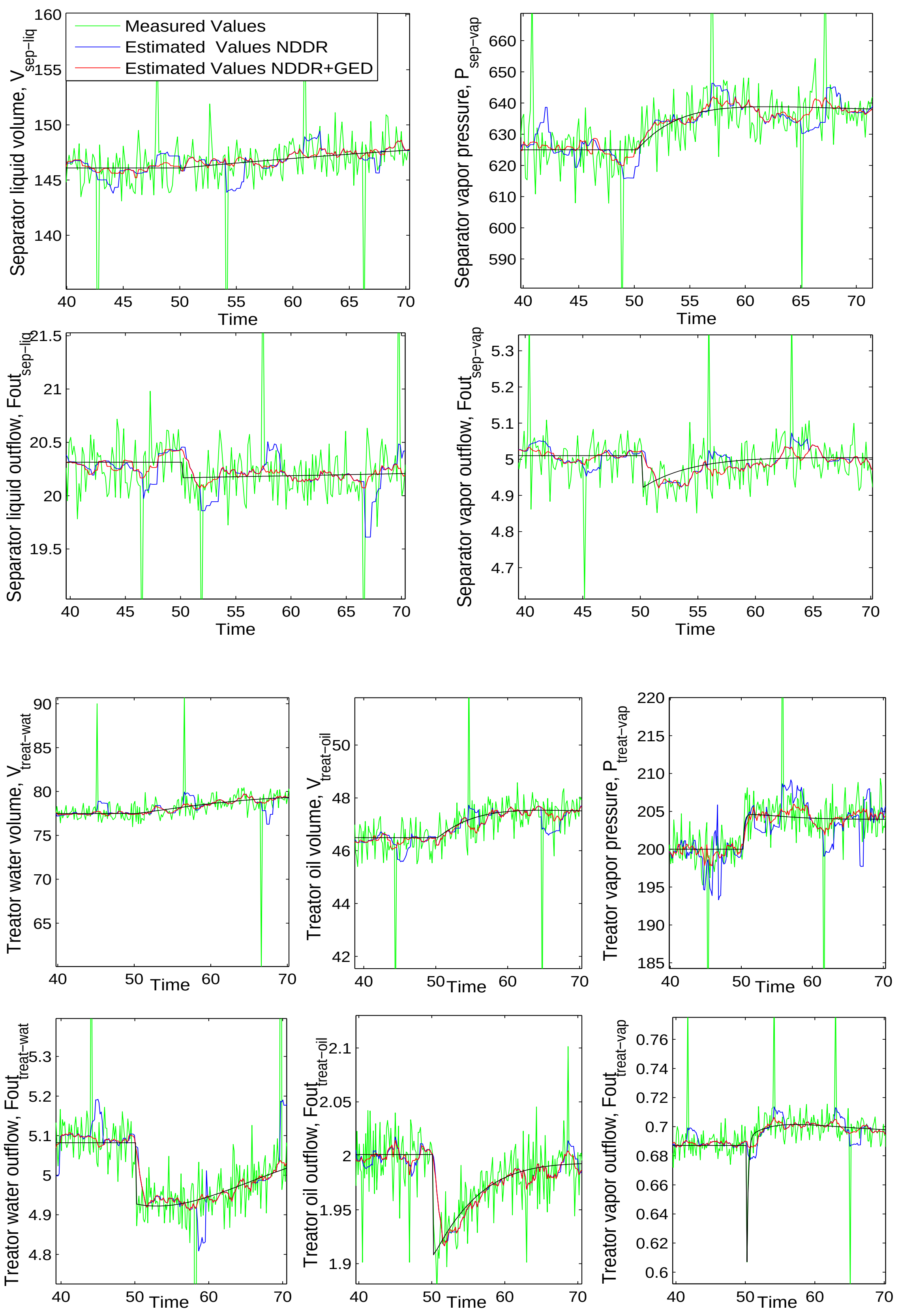


Figure 6.2: ANDDR + GEDR results with positive set point change

A quantitative comparison between the noisy measurements, the basic NDDR stimates and the ANDDR + GEDR estimates confirming the qualitative results mentioned above is presented in table 6.1. In every variable, GEDR produces a significant decrease in noise standard deviation.

		% Noise reduction		% RMSE reduction	
	Variable	NDDR	ANDDR+GEDR	NDDR	ANDDR+GEDR
Inputs	$Fout_{sep-liq}$	69.72	83.67	62.93	83.40
	$Fout_{sep-vap}$	69.06	78.58	67.37	71.21
	$Fout_{treat-wat}$	78.09	90.73	69.79	88.64
	$Fout_{treat-oil}$	65.72	69.12	55.32	58.63
	$Fout_{treat-vap}$	65.60	85.96	68.49	85.96
Outputs	$V_{sep-liq}$	68.51	85.96	68.49	85.96
	$P_{sep-vap}$	69.47	83.12	63.33	81.94
	$V_{treat-wat}$	77.76	88.70	73.14	88.65
	$V_{treat-oil}$	72.84	84.67	68.70	84.73
	$P_{treat-vap}$	49.91	82.19	43.71	80.09

Table 6.1: Comparison between NDDR and ANDDR+GEDR

This study was performed to confirm the value of GEDR, as proposed in [1]. The approached was tested, and refined in order to make it more efficient to work in a complex model such as the pilot plant.

Chapter 7

Thesis Observations

Previous studies implemented DR and GED in academical models such as the jacketed continuous stirred tank reactor. The focus of the present research was to extend previous work on these techniques to implement them in a complex and realistic model such as the pilot plant. Several problems that were not seen with academic models, were presented, faced, and solved. Different tests were performed with the NDDR algorithm presented by Laylabadi and Taylor [1] [4]. As a result, important conclusions were obtained, which helped to refine and to implement more efficiently the technique into the pilot plant model. A new hybrid approach was proposed and tested. This new approach allows the technique to be applied into systems that do not have a dynamic nonlinear model available. A SSD algorithm was developed and implemented in the pilot plant as well. The algorithms developed in this thesis are compatible with the ICAM System [5].

In this chapter a summary of the most important conclusions and contributions attained from this thesis are presented, and future work is proposed.

7.1 Conclusions

The following conclusions can be drawn from the work presented in this thesis:

1. An algorithm for steady-state detection was developed and successfully tested on the pilot plant model using different scenarios. The algorithm can accurately determine when a single variable and/or the complete system are in transient or steady state conditions.

2. The steady-state algorithm was implemented as an agent which is compatible to work in conjunction with the smart supervisory system [5]. The algorithm is able to continually inform the supervisor about the state of the variables and the state of the system overall.

3. Statistical methods fail at detecting steady-state for the pilot plant model. This is due to the constraints imposed by the nonlinear model. The amount of noise and set-point change allowed by the nonlinear model are limited and in some cases these values can be very close. When the set-point change and the standard deviation of the noise are similar, the statistics of the data cannot provide a pattern that permit differentiating transient from steady state. The method developed in this thesis can accurately establish the state of the plant even when the set point change and the standard deviation of the noise are equal.

4. The NDDR algorithm presented by Laylabadi and Taylor [1] [4] was implemented and tested in the pilot plant model. The outcome of the different test show that their approach used to tackle the optimization problem for the continuous stirred tank reactor (CSTR) or the Jacketed CSTR (JCSTR) models is not efficient when it is applied to a more complex model such as the pilot plant model.

5. The ANDDR algorithm was refined by using a large-scale optimization algorithm. The optimization routine used is based on the interior-reflective Newton method and it requires the gradient of the objective function to perform the minimization. The NDDR implemented with the large-scale algorithm provide a robust, efficient and reliable data reconciliation for the separator model.

6. Applying the ANDDR technique to such a complex model as the pilot plant is very expensive timewise. In order to reduce the computation time a hybrid approach was developed and tested. This algorithm uses different methods to perform the data reconciliation depending of the plant state (transient or steady state). The effectiveness of the hybrid algorithm to reduce the computation time depends on the availability of a simpler model for the steady state interval.

7. There are different parameters that affect the performance of the data reconciliation algorithm. These parameters include the size of the history window, the use of scaling, the initial guesses at the optimal solution, and the tolerance of the optimization routine. The selection of these parameters was studied to determine the effect on the efficiency of the algorithm.

8. The size of the history window is an important tuning parameter for the optimization routine. This parameter allows making a compromise between the amount of noise reduction required and the computation time spent.

9. Good initial guesses at the optimal solution for the minimization problem can help to obtain a faster optimization. The best performance was obtained when the estimates at the previous step were used as initial guesses.

10. Using the large-scale optimization routine, the ANDDR and GEDR algorithm were implemented on the separator model. Gross errors were detected and removed. The reconciliation was performed successfully in scenarios with un-

known noise statistical model.

11. The ANDDR + GEDR algorithms were implemented as an agent compatible with the ICAM supervisor system.

7.2 Future Work

There are two important steps to continuing with the developments undertaken in this study. The first step is to develop a simpler steady state model based on material and energy balance to be used as part of the hybrid algorithm. Using a simpler model will help to reduce the computation time, and it will allow running the data reconciliation algorithm in real time.

The second step will be the implementation of the two agents presented in this thesis using the actual data from the pilot plant facility at the College of North Atlantic (CNA), which is part of the Petroleum Application of Wireless System (PAWS) project. This will allow the SSD and ANDDR + GEDR agents to be tested in a real industrial application.

Bibliography

[1] M. Laylabadi and J. H. Taylor. Anddr with novel gross error detection and smart tracking system. *12th IFAC Symposium on Information Control Problems in Manufacturing (INCOM2006), Saint-Etienne, France*, May 2006.

[2] A. Sayda and J. H. Taylor. Modeling and control of three-phase gravity separators in oil production facilities. *Proc. American Control Conference, New York, USA*, 2007.

[3] M. J. Liebman, T. F. Edgar, and L. S. Lasdon. Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques. *Computers chem. Engng*, 16(10/11):963–986, 1992.

[4] J. H. Taylor and M. Laylabadi. A novel adaptive nonlinear dynamic data reconciliation and gross error detection method. *IEEE Conference on Control Applications, Munich, Germany*, October 2006.

[5] J. H. Taylor and A. Sayda. Prototype design of a multi-agent system for integrated control and asset management of petroleum production facilities. *Proc. American Control Conference, Seatle, Washington*, June 2008.

[6] D. R. Kuehn and H. Davidson. Computer control ii: Mathematics of control. *Chem. Eng. Prog.*, 57(44), 1961.

[7] G. M. Stanley R. S. H. Mah and D. M. Downing. Reconciliation and rectification of process flow and inventory data. *Ind. Eng. Chem. Process Des. Dev.*, 15(1):175–183, 1976.

[8] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, pages 35–45, March 1960.

[9] A. Gelb. Applied optimal estimation. *MIT Press, Cambridge, MA*, 1974.

[10] G. M. Stanley and R. S. H. Mah. Estimation of flows and temperatures in process networks. *AIChE Journal*, 23:642–650, 1977.

[11] G. A. Almasy. Principles of dynamic balancing. *AIChE J.*, 36(9):1321–1330, 1990.

[12] J. H. Lee D. G. Robertson and J. B. Rawlings. A moving horizon-based approach for least squares estimation. *AIChE J.*, 42(8):2209–2224, 1996.

[13] Louis P. Russo Tyler A. Soderstrom, Thomas F. Edgar and Robert E. Young. Industrial application of a large-scale dynamic data reconciliation strategy. *Ind. Eng, Chem. Res.*, 39:1683–1693, 2000.

[14] J. C. Knepper and J. W. Gorman. Statistical analysis of constrained data sets. *AIChE J.*, 26(2):260–264, 1980.

[15] B. Joseph S. S. Jang and H. Mukai. Comparison of two approaches to on-line parameter and state estimation of nonlinear systems. *Ind. Engng Chem. Process. Des. Dev.*, 25:809–814, 1986.

[16] M. J. Liebman and T. F. Edgar. Data reconciliation for nonlinear process. *Proceedings of the AIChE Annual Meeting, Washington, DC*, 1998.

[17] J. A. Romagnoli G. H. Weiss and K. A. Islam. Data reconciliation - an industrial case study. *Computers chem. Engng*, 20(12):1441–1449, 1996.

[18] Thomas F. Edgar Karen F. McBrayer, Tyler A. Soderstrom and Robert E. Young. The application of nonlinear dynamic data reconciliation to plant data. *Computers Chem. Engng*, 22(12):1907–1911, 1998.

[19] J. Placido and L.V. Loureiro. Industrial application of data reconciliation. *Computers chem. Engng*, 22:S1035–S1038, 1998.

[20] G. A. Almasy and B. Uhrin. Principles of gross measurement error identification by maximum-likelihood-estimation. *Hungarian J. of Industrial Chemistry*, 21:309, 1993.

[21] D. L. Ripps. Adjustment of experimental data. *Chem. Engng. Prog. Symposium*, 61:8–13, 1965.

[22] P. M. Reilly and R. E. Carpani. Application of statistical theory of adjustment to material balances. 13^{th} *Can. Chem. Eng. Conf., Montreal, Quebec*, 1963.

[23] G. A. Almasy and T. Sztano. Checking and correction of measurements on the basis of linear system model. *Problems of Control and Information Theory*, 4:57–69, 1975.

[24] V. Veverka F. Madron and V. Vanecek. Statistical analysis of material balance of a chemical reactor. *AIChE J*, 23:482, 1977.

[25] Shankar Narasimhan and Cornelius Jordache. *Data Reconciliation and Gross Error Detection: An Intelligent Use of Process Data.* Gulf Professional Publishing, 2000.

[26] S. Narasimhan and R. S. H. Mah. Generalized likelihood ratio method for gross error identification. *AIChE J.*, 33:1514, 1987.

[27] H. Tong and C. M. Crowe. Detection of gross errors in data reconciliation by principal component analysis. *AIChE J.*, 41:1712, 1995.

[28] D. M. Himmelblau T. A. Soderstrom and T. F. Edgar. A mixed integer optimization approach for simultaneous data reconciliation and identification of measurement bias. *Control Engineering Practice*, 9:869–876, 2001.

[29] Shrikant A. Bhat and Deoki N. Saraf. Steady state identification, gross error detection, and data reconciliation for industrial process units. *Ind. Eng. Chem. Res.*, 43:4323–4336, 2004.

[30] S. Narsimhan, R. S. H. Mah, A. C. Tamhane, J. W. Woodword, and J. C. Hale. A composite statistical test for detecting changes of steady states. *AIChE J.*, 32:1409–1418, 1986.

[31] S. Narsimhan, C. N. Kao, and R. S. H. Mah. Detecting changes of steady state using the mathematical theory of evidence. *AIChE J.*, 33(11):1930–1933, 1987.

[32] J. Loar. Control for the process industries. *Putman Publications, Chicago, IL*, 7(11):62, November 1994.

[33] S. L. Alekman. Control for the process industries. *Putman Publications, Chicago, IL*, 7(11):62, November 1994.

[34] G. Jubien and G. Bihary. Control for the process industries. *Putman Publications, Chicago, IL*, 7(11):64, November 1994.

[35] S. Cao and R. R. Rhinehart. An efficient method for on-line identification of steady state. *J. Process Control*, 5(6):363–374, 1995.

[36] A. Sayda and J. H. Taylor. An implementation plan for integrated control and asset management of petroleum production facilities. *Procc. IEEE Conference on Control Applications (CCA2006)*, 2006. Munich, Germany.

[37] A. Sayda and J. Taylor. A multi-agent system for integrated control and asset management of petroleum production facilites - part 1: Prototype design and

develpoment. *IEEE International Symposium on Intelligent Control*, September 2008. San Antonio, Texas.

[38] M. Omana. *Fault Detection, Isolation and Accomodation Using the Generalized Parity Vector Technique.* PhD thesis, University of New Brunswick, 2009.

[39] J. H. Taylor and M. Omana. Fault detection, isolation and accomodation using the generalized parity vector technique. *Proc. IFAC World Congress, Seoul, Korea*, July 2008.

[40] H. M. S. Ibrahim. Wireless sensor network management agent. Proposal, University of New Brunswick, 2009.

[41] MATHWORKS.COM. Optimization toolboxfor use with MATLAB®, 1990-2004.

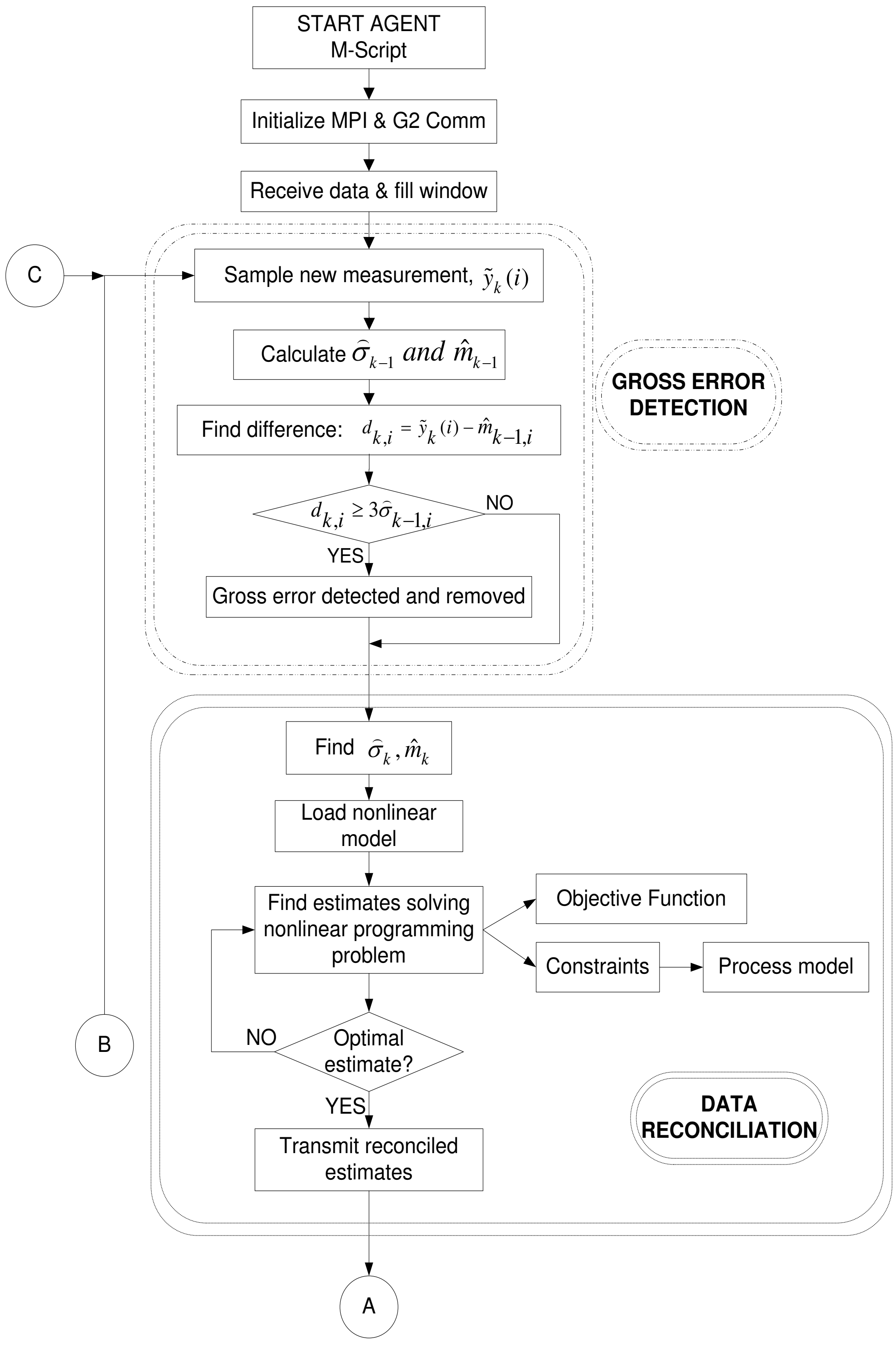
[42] T.F. Coleman and Y. Li. An interior, trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6:418–445, 1996.

[43] J. Taylor. *EE 4323 Industrial Control Systems.* Department of Electrical and Computer Engigeering - University of New Brunwick, 2007.

Appendix A

ANDDR+GED and SSD Interaction with Multi-agent Supervisory System

The figure presented in this appendix illustrates the communication and data exchange process between the ANDDR + GED Agent, Steady-state Agent and other agents in the system, as discussed in chapter 2. This shows how my contributions fit into the large and complex ICAM system.



START AGENT
M-Script
Initialize MPI & G2 Comm
Receive data & fill window
C
Sample new measurement, $\tilde{y}_k(i)$
Calculate $\hat{\sigma}_{k-1}$ and $\hat{m}_{k-1}$
GROSS ERROR DETECTION
Find difference: $d_{k,i} = \tilde{y}_k(i) - \hat{m}_{k-1,i}$
$d_{k,i} \geq 3\hat{\sigma}_{k-1,i}$
NO
YES
Gross error detected and removed
Find $\hat{\sigma}_k, \hat{m}_k$
Load nonlinear model
Find estimates solving nonlinear programming problem
Objective Function
Constraints
Process model
B
NO
Optimal estimate?
YES
Transmit reconciled estimates
DATA RECONCILIATION
A

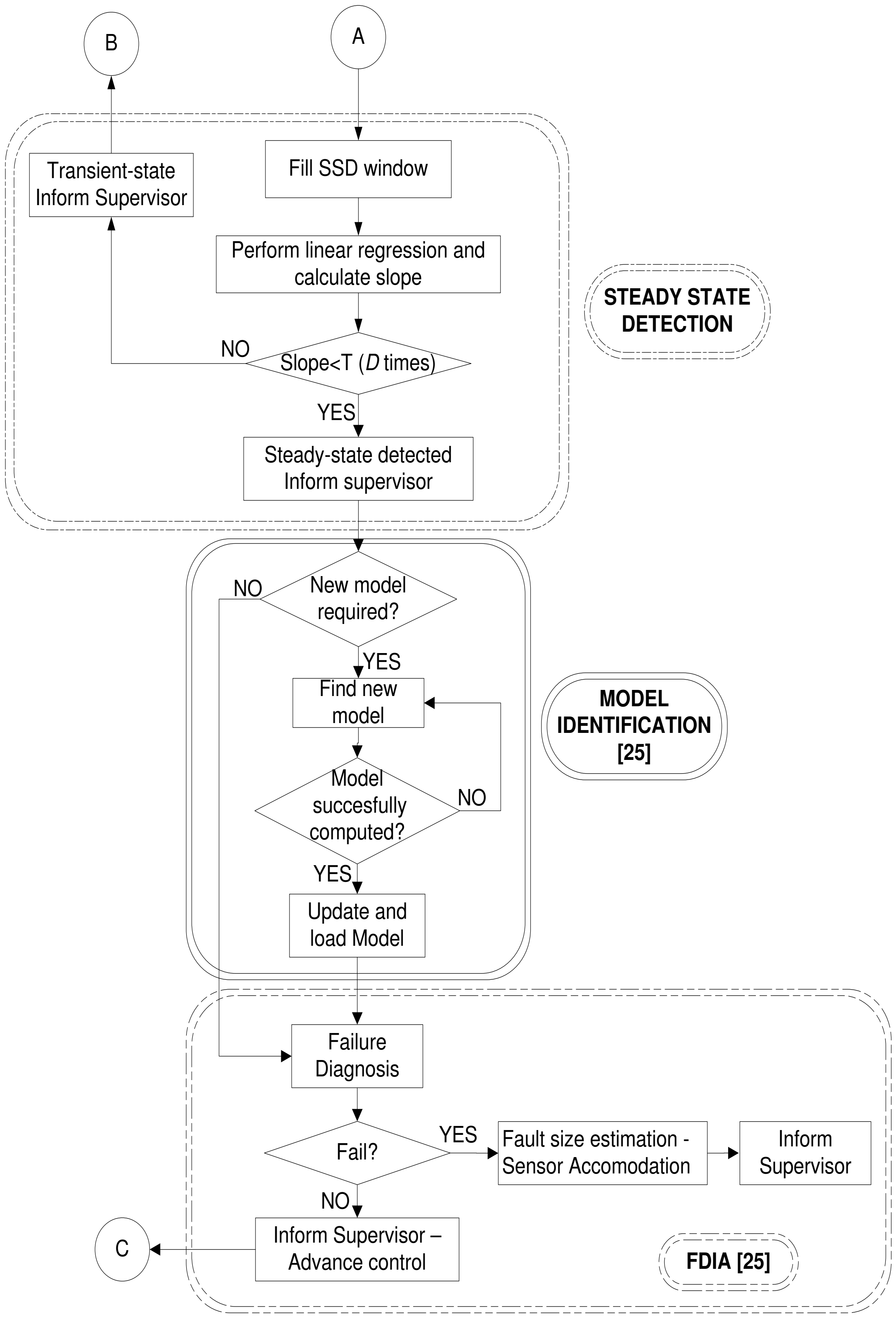

B
A
Transient-state Inform Supervisor
Fill SSD window
Perform linear regression and calculate slope
STEADY STATE DETECTION
NO
Slope<T (D times)
YES
Steady-state detected Inform supervisor
NO
New model required?
YES
Find new model
MODEL IDENTIFICATION [25]
Model succesfully computed?
NO
YES
Update and load Model
Failure Diagnosis
YES
Fail?
Fault size estimation - Sensor Accomodation
Inform Supervisor
NO
C
Inform Supervisor – Advance control
FDIA [25]

Figure A.1: Interaction ANDDR + GED and SSD with the Multi-agent Supervisory System.

Appendix B

Simulation Model

The simulation model used for developing and testing the two algorithms presented in this thesis consists of a two-phase separator followed by a three-phase separator or treator. Gravity separators are hydrodynamic separation mechanisms used to remove grit, heavy sediments, grease, debris and floatable matter from water and crude oil through gravitational settling and trapping. In oil production facilities the function of a separator is to divide the oil well stream into either two phases (gas and liquid streams) or three phases (gas, crude oil and water). A separator works based on the relatively low solubility of petroleum products in water and the difference between the specific gravity of water and the specific gravity of petroleum compounds.

A nonlinear model and control system for the two phase separator followed by three phase separator were developed by Sayda and Taylor [2]. This model corresponds to the real pilot plant located at the College of North Atlantic (CNA) and was chosen as a test bed for the PAWS project. This chapter describes the general operation of a gravity separator, and the model implemented by Sayda and Taylor.

B.1 Process Description

A two-phase separator operates on the same principles as a three-phase one, except the liquid component is not separated into crude oil and water. We will focus here on the more complex three-phase separator. The function of a three-phase separator is to divide the water from the mixture of oil and water. A basic scheme of a separator is shown in figure B.1.

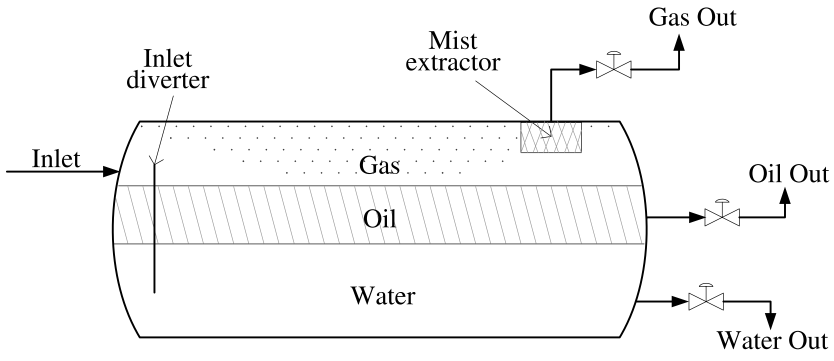


Figure B.1: Three phase separator scheme

The fluid enters the device and collides with an inlet diverter; in this part of the process occurs the first coarse separation of liquid and vapor. The inlet diverter minimizes the amount of gas that remains in the liquid and assures that the liquid is not going to flow above the gas/oil or oil/water interface. The vapor flows over the inlet diverter to settle down above the liquid section. The gas may contain small drops of liquid, which are then separated by gravity and fall to the gas-liquid interface. A mist extractor is used to remove the remaining liquid from the gases before they are pumped out.

B.2 Gravity Separator Mathematical Model

The purpose of this section is to show the three phase gravity separator mathematical model developed by Sayda and Taylor. A brief explanation of the model and the equations are provided. For a complete description refer to [2].

The three phases of the separator are: Aqueous, oil and gas. The phases and the different streams flowing between them are shown in figure B.2.

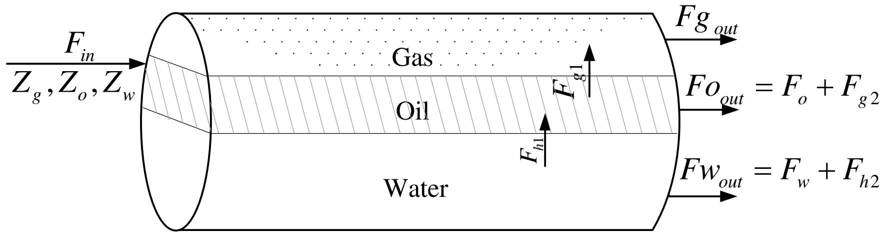


Figure B.2: Separated component streams

The fluid injected into the separator F_{in} is a mixture of gas, oil and water with molar fractions of Z_g, Z_o and Z_w respectively. The hydrocarbon component is divided in two different streams; F_{h1}, which separates by gravity and goes directly to the oil phase and F_{h2} that remains in the aqueous phase due to incomplete separation. The gas component is divided as well into two streams F_{g1} and F_{g2}; the former flows out of the oil phase due to the pressure drop in the separator, and the second stays dissolved in the oil phase. The dynamics of each phase of the separator were modeled separately.

B.2.1 The Aqueous Phase

The dynamic material balance equations of the aqueous phase are:

$$\begin{aligned}
F_{h1v} &= \frac{\varepsilon(Z_g + Z_o)F_{in}Mw_h}{62.43SG_h} \\
F_{h2v} &= \frac{(1-\varepsilon)(Z_g + Z_o)F_{in}Mw_h}{62.43SG_h} \\
Fw_{out} &= \frac{Z_w F_{in} Mw_w}{62.43SG_w} + F_{h2v} \\
\frac{dV_{wat}}{dt} &= \frac{F_{in}Mw_{in}}{62.43SG_{in}} - Fw_{out} - F_{h1v}
\end{aligned}$$

In the previous equations, ε is the unseparated hydrocarbon fluid volume fraction, Mw_h, Mw_w and Mw_{in} are the molecular weights for the hydrocarbon, water and incoming mixture, F_{h1v} and F_{h2v} are the separated and unseparated volumetric flow components of the hydrocarbon fluid, and SG_h, SG_w and SG_{in} correspond to the specific gravities. V_{wat} is the aqueous phase volume; and Fw_{out} is the water discharge volumetric outflow.

B.2.2 The Oil Phase

The separated hydrocarbon stream (F_{h1}) is composed by gas and oil (Z_{g1}, Z_{o1}). A part of the gas is flashed to the gas phase and the other part remains in the oil phase. Using the following equations is possible to find the quantity of flashing gas F_{g1}, the amount of gas that stays dissolved in the oil phase F_{g2} and the oil discharge flow Fo_{out}:

$$\begin{aligned} x &= P/P_v \\ F_{g1} &= (1-x)Z_{g1}F_{h1} \\ F_{g2} &= xZ_{g1}F_{h1} \\ Fo_{out} &= F_o + F_{g2} \\ \frac{dN_{oil}}{dt} &= F_{h1} - F_{g1} - Fo_{out} \\ Mw_{O1} &= xMw_g + (1-x)Mw_o \\ SG_{O1} &= \frac{xMw_gN_{oil} + (1-x)Mw_oN_{oil}}{\frac{xMwgNoil}{SGg} + \frac{(1-x)MwoNoil}{SGo}} \end{aligned}$$

where x is the mole fraction of gas into the liquid phase, P is the total pressure of the vapor phase, P_v is the vapor pressure of the gas, Z_{g1} is the gas molar fraction, N_{oil} is the number of liquid moles in the oil phase, F_o is the molar oil component in the oil discharge flow Fo_{out}, Mw_g and Mw_o are the gas and oil molecular weights and finally SG_g and SG_O are the gas and oil specific gravities.

B.2.3 The Gas Phase

In the gas phase it is assumed that the gas in the vapor phase is an ideal gas. The gas pressure is estimated from the ideal gas law:

$$
\begin{aligned}
\frac{dN_{gas}}{dt} &= F_{g1} - F_{g_{out}} \\
V_{oil} &= \frac{Mw_{o1}N_{oil}}{62.43SG_{o1}} \\
V_{gas} &= V_{sep} - V_{wat} - V_{oil} \\
P &= \frac{N_{gas}RT}{V_{gas}}
\end{aligned}
$$

where N_{gas} is the number of gas moles in the gas phase, Fg_{out} is the gas molar outflow from the separator, V_{oil}, V_{gas} and V_{sep} are the volumes of the oil phase, gas phase and separator respectively. R is the universal gas constant and T is the absolute separator temperature.

B.3 Pilot Plant Model

The plant model used in this thesis as the test bed simulation model is based on the three-phase gravity separator as explained in section B.2, preceded by a two-phase separator. The simulation model schematic is shown in figure B.3. The plant can be observed as two different processes. The first part is a two-phase separator which divides hydrocarbon fluids extracted from oil wells into two phases, gas and oil + water. This separator is $15ft$ long and has a diameter of $5ft$. To model this part of the process one takes into account just the liquid and gas phases explained before.

The second process is a three-phase separator or treator, which takes the liquid produced previously and separates water and solids from oil. The oil produced is then pumped out and sold to refineries and petrochemical plants. The length of the three-

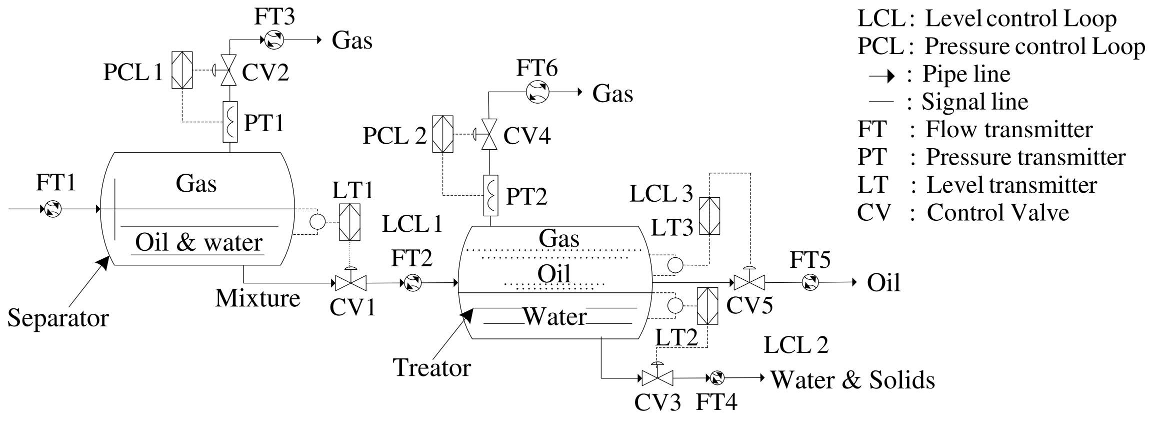


Figure B.3: Pilot Plant model, comprised of a separator and a treator

phase separator is $8.6ft$ and a diameter of $4.8ft$.

Both separation processes are controlled to operate at their nominal values. The first separation process uses two PI controller loops to maintain the liquid level and the pressure inside the two-phase separator. The second process has three PI controller loops to maintain the level of water/oil interface, the level of oil/gas interface and the vessel pressure [2].

Appendix C

SSD Algorithm Results

Figure C.1 illustrates the application of the SSD algorithm to the pilot plant model in the scenario where the setpoint change and noise standard deviation are equal. In this example the setpoint change is applied at time $t = 0$ *sec.* for all the variables. The percentage of change in the setpoint is 2%, as is the σ. It is important to observe that the method adopted to establish steady-state works properly even when the change in the variable is the same as the noise.

Figure C.2 shows the performance of the algorithm for an scenario with positive and negative set point changes. The plant starts at nominal operating point, and at the time $t = 100$ *sec.* a set point change of -5% is applied to $V_{sep-liq}$. Afterwards, at time $t = 450$ *sec.*, $P_{sep-vap}$ has a set point variation of $+10\%$. Later, at time $t = 500$ *sec.*, there is a set point change of -2% in $V_{treat-wat}$. Subsequently, $V_{treat-oil}$ has a set point change of $+10\%$ at time $t = 550$ *sec.*, and finally at time $t = 650$ *sec.* there is a set point variation of -10% in $P_{treat-vap}$.

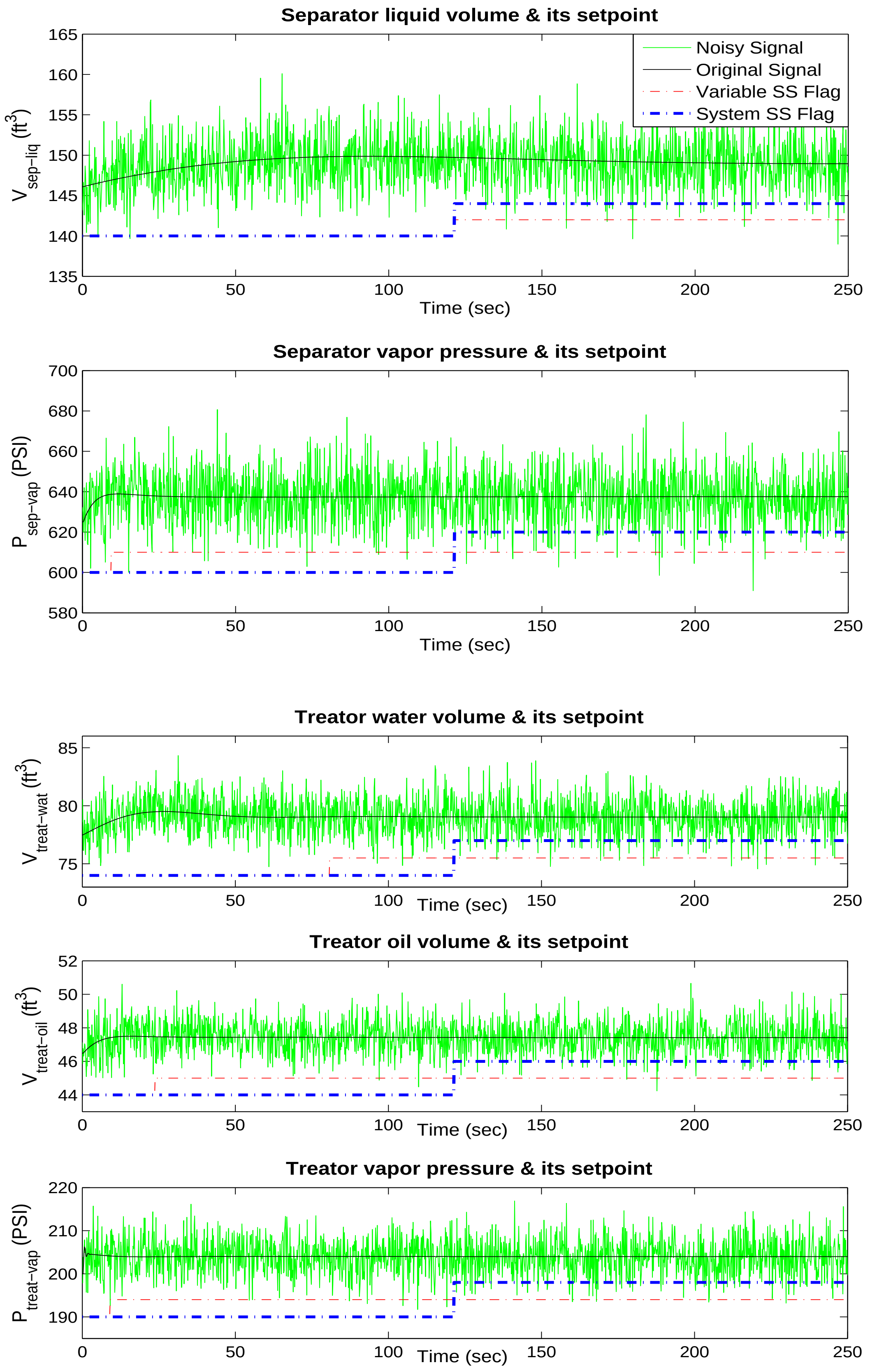


Figure C.1: SSD Separator Outputs - Setpoint change = Noise standard deviation

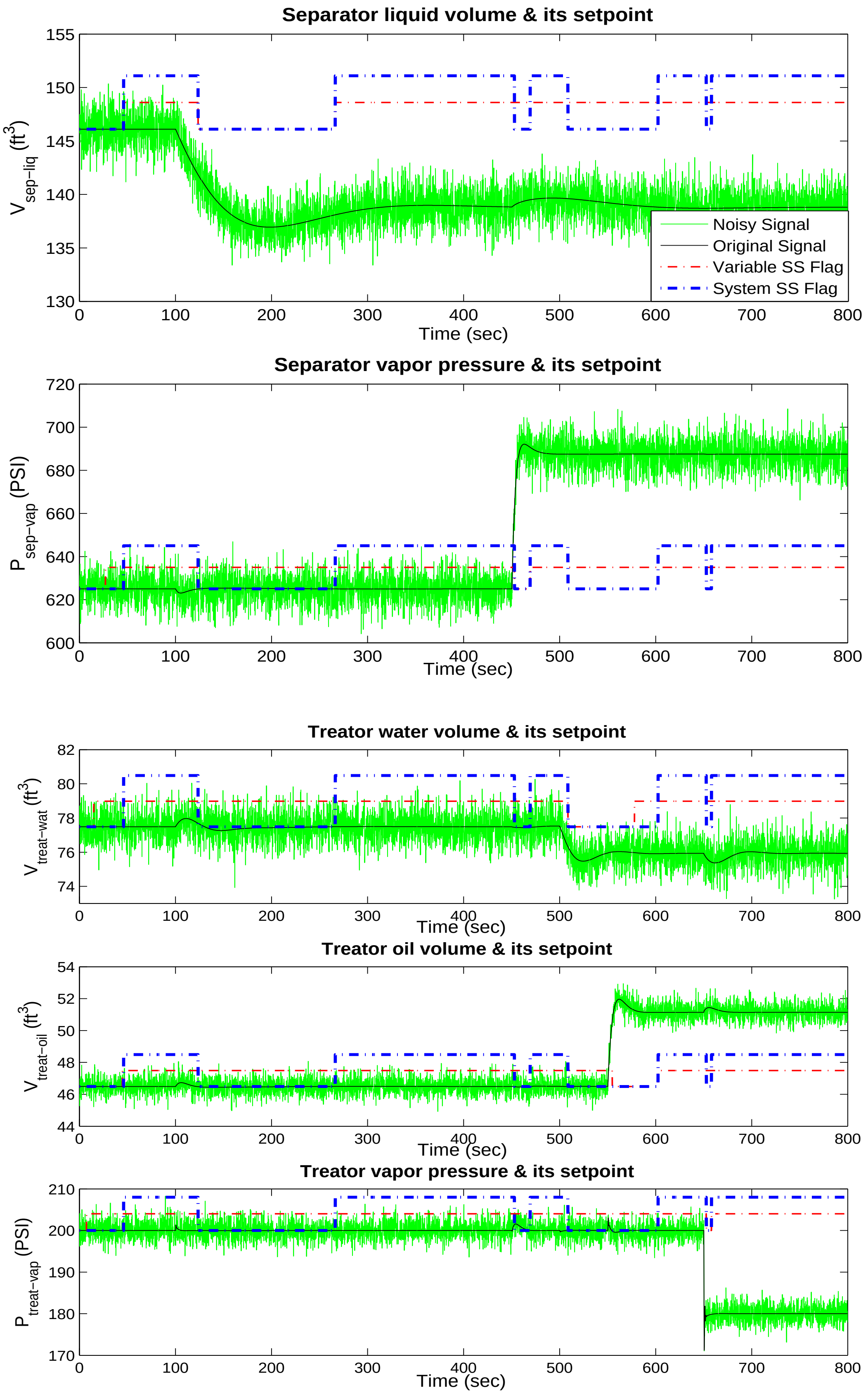


Figure C.2: SSD Separator Outputs - Positive and Negative Setpoint change

Appendix D

Comparison between NDDR and low pass filtering

Figures D.1 and D.2 show the pilot plant response to the NDDR and to the low pass filter for the variables $Fout_{treat-wat}$, and $Fout_{treat-oil}$. It can be observed that there is a significant dynamic lag presented in the filtered signal.

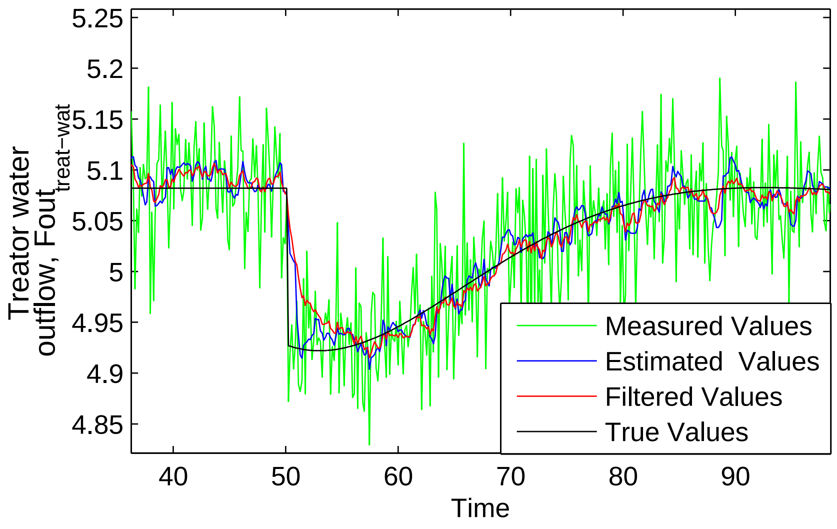


Figure D.1: Comparison NDDR Vs Filter for $Fout_{treat-wat}$

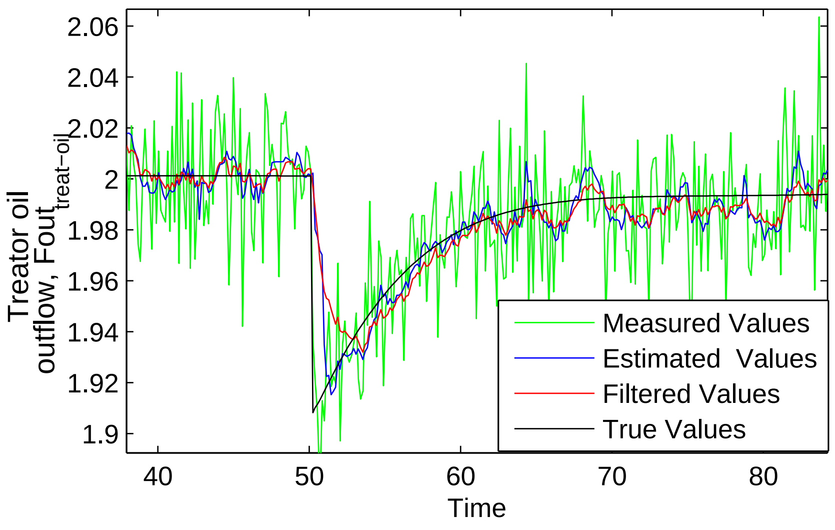


Figure D.2: Comparison NDDR Vs Filter for $Fout_{treat-oil}$

Vita

Candidate's full name:	Rocio del Pilar Moreno Viancha
Universities attended:	Universidad Industrial de Santander, Bachelor in Electrical Engineering, 2001
	Universidad Distrital Francisco Jose de Caldas, Master of Engineering in Mobile Communications, 2004