# Fault Detection, Isolation and Remediation of Real Processes

by

B M Sirajeel Arifin

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Process Control

Department of Chemical and Materials Engineering

University of Alberta

# Abstract

The goal of process control and monitoring system is to operate the process plants safely, reliably and in fault-free mode. Even a small fault in the process operation or in the control logic can reduce the efficiency of the process greatly. Normal process transients and operating condition changes can affect the process monitoring and control system adversely if the system cannot identify and separate these normal condition changes from the faults. Such situations may cause a lot of false alarms or alarm-floods and operators may lose confidence in these systems. Therefore, an innovative data-driven change detection algorithm has been proposed. This novel approach will enable the operators to detect changes from the normal condition. The proposed change detection algorithm is based on the Kantorovich distance concept and it does not depend on the type of statistical distribution of the data. The method was applied successfully to the benchmark Tennessee Eastman process (TEP) to detect both abrupt and ramp type changes.

Inherent to both the process and pipeline industries are specific faults. In this study, efficient detection and remedial algorithms have been addressed for two such faults. The first fault addressed in this research is the pipeline leak which can cause serious environmental and societal/public safety issues. Such faults can often lead to huge environmental cleanup fees, penalties plus the operation downtime and product losses and poor media coverage. The existing leak detection methods perform poorly in pinpointing the leak location when leak size is less than 10% of the nominal flow-rate. When the leak size is less than 5%, the existing methods take 0.5 to 1 hr or more to detect the leak. To alleviate these limitations, a novel data-driven leak detection and localization method have been proposed in this study. Unlike the model-based methods, this method does not require tedious parameter tuning. Industrial evaluation of the proposed method in real-time during regulatory leak tests at Suncor Pipeline has confirmed the efficacy of the method. The proposed method

was successful in detecting and localizing leak as large as 7% to as small as 1%. The leak localization efficiency was comparable with the existing commercial method. The leak detection time was much faster than the existing method.

Another fault that is commonly encountered in process industries is a sticky actuator. The control valve is the main actuating element in the control loops for both process and pipeline industries. Control valve stiction is the main cause of sustained limit cycles in most control loops and causes unwanted product variability, off-spec product, loss of energy and raw materials. So, early detection of stiction is a primary concern. A novel data-driven stiction detection scheme is proposed in this thesis based on the Kantorovich distance methodology.

Since stiction is a physical problem, to replace or repair the valve needs an unplanned shutdown of the plant. Stiction compensation algorithm can reduce the effect of stiction till the next planned shutdown. A novel stiction compensation algorithm has been proposed in this research where a variable amplitude compensating signal is added to the control signal to move the valve stem from its sticky position. This variable amplitude compensating signal can handle uncertainty in the estimation of the stiction parameter. The proposed compensation scheme was applied successfully to a pilot plant equipped with a real sticky valve including built-in valve positioner. The method was successful for both flow and level control loops. The method can reduce the effect of the limit cycle, track dynamic setpoint changes automatically and reject disturbances with less amount of valve movement than other traditional methods.

# Preface

The materials presented in this thesis are part of the research project under the supervision of Dr. Sirish Shah and Dr. Zukui Li, and is funded by Natural Sciences and Engineering Research Council (NSERC) of Canada, Suncor and Syncrude.

Chapter 2 of this thesis is an extended version of the accepted manuscript for the proceedings of $10^{th}$ IFAC symposium on Advanced Control of Chemical Processes (ADCHEM) to be held on 2018 at Shenyang, China. The proposed title for this conference paper is 'Change Point Detection using the Kantorovich Distance Algorithm'.

Chapter 4 of this thesis is an extended version of 'Pipeline Leak Detection Using Particle Filters' (Arifin et al., 2015) in the proceedings of $9^{th}$ IFAC symposium on Advanced Control of Chemical Processes (ADCHEM) held on 2015 at Whistler, British Columbia, Canada.

Chapter 3 and 5 are revised version of 'A Novel Data-Driven Leak Detection and Localization Algorithm using the Kantorovich Distance',2018, 'Computers & Chemical Engineering', Volume 108, pages 300-313 (Arifin et al., 2018a). This work was done in collaboration with Suncor Pipeline. Suncor Pipeline facilitated the industrial evaluation of the Data-Driven Leak Detection Approach.

Chapter 6 of the thesis is an extended version of 'A Model Free Approach for Online Stiction Compensation' (Arifin et al., 2014) in the proceedings of $19^{th}$ IFAC World Congress (2014) held in Cape Town, South Africa.

Chapter 7 of the thesis is a revised version of 'Actuator Stiction Compensation via Variable Amplitude Pulses', 2018, 'ISA Transactions®', Volume 73, pages 239-248 (Arifin et al., 2018b). This work was done in collaboration with Dr. Celso Munaro of Departamento de Engenharia Eltrica, UFES, Vitria, ES, Brazil. The algorithm was tested in the pilot plant at Brazil which was equipped with industrial size sticky valve.

# Acknowledgements

Foremost I want to thank the Almighty Allah who has created the whole universe including human beings, all the animals and plants. He has given us the knowledge, strength and good health which are the primary necessity for accomplishing any work. I would also like to extend my gratitude to my parents who have struggled hard from my birth to keep me happy and make my future prosperous. I should also thank my wife, who was an inspiration and assisted me, despite completing her own Ph.D. studies throughout the course of my Ph.D. My two years old son also deserves acknowledgment because after a long day his bright laugh removed the long hours and tiredness.

I would like to express my heartiest gratitude to my supervisors Dr. Shah and Dr. Zukui for being patient during the hard times, for providing useful guidance, freedom and continuous support. Without their support, this massive task would not be feasible. I would also appreciate the financial support of Dr. Tongwen Chen and Dr. Shah and NSERC, Suncor and Syncrude to conduct this research work.

I am also grateful to Suncor pipelines for the pipeline data they provided for leak detection study. They have also provided me with the opportunity to implement my data-driven leak detection method to one of their pipelines real-time during the annual regulatory leak tests. I would like to specially mention the name of Gordon and Amanda of Suncor pipelines for their kind support. Dr. Prakash helped me to understand the theory of particle filter. Dr. Yuntong She helped me to understand the hydraulic behavior in the pipeline during a leak.

Dr. Celso Munaro and his team helped to implement the variable amplitude stiction compensation method in the pilot plant in Brazil. I would like to thank Petrobrass for providing the industrial valve on which all the experimental works related to this work were performed and the engineer Fernando Tadeu Rios Dias for his valuable support on

valve positioner configuration using asset management software.

The preliminary version of the stiction compensation method was implemented in CPC laboratory of the University of Alberta. I would like to extend my thanks to lab coordinator of that time Dr. Yuri and lab technicians for their kind help. My group-mate Sridhar helped me to connect my algorithm to the DeltaV DCS system and Dr. Shoukat helped with his extensive expertise in the field of control valve stiction.

Finally, the group members of 'Advances in Alarm Management and Design (AAMD)' and 'Optimization and System Engineering' were very supportive throughout my Ph.D. study. I have been very lucky to be part of these two strong, highly motivated and kind groups.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 What is a Fault?

A fault in a process plant or pipeline can be described as an unacceptable or undesired deviation of the process-variables which can be detected by an appropriate signal evaluation (Martinez-Guerra and Mata-Machuca, 2016). In other words, when a system shows an unpermitted deviation from the acceptable or standard condition of the process, then it is said that the system is faulty. It represents the malfunction of the system or a system component (Venkatasubramanian et al., 2003c). Extreme cases of faults are termed as 'failures' and may cause catastrophic accidents (Baillieul and Samad, 2015).

A process may fail due to leaks or defects in a pipeline or vessel; malfunction of pumps, compressors and other rotary equipment and valve; parameter uncertainty in the process models; failure of energy supply; off-spec or shortage of raw materials; production of undesired product and so on (Raghavan, 2004). Constrained resources, stringent environmental regulations and tough competition between different industries have resulted in highly efficient manufacturing operations in terms of energy and raw material utilization, optimal quality products and safety of the plant personnel and surrounding communities all with a lower cost. Most of the modern plants are now automated to achieve these goals. The process control and monitoring system is an essential part of these automated processes and in large process plants, there are as many as hundreds and even thousands of such control loops (Arifin et al., 2014).

A typical control loop consists of sensor, transmitter, I/P (current to pressure) converter, P/I(pressure to current) controller, actuator, input-output devices. Alarms are also raised by this system when an abnormality in the process is observed. Fault in any of these instruments or connection between two instruments or miscommunication, unknown disturbances or inaccurate tuning of the controller parameters can make the control system unreliable. Performance of over 26,000 proportional-integral-derivative (PID) controllers

from a wide range of continuous process industries have been investigated (Desborough and Miller, 2002) and it is shown that the performance of over two-thirds of these installed loops (68% to be precise) was not satisfactory (Figure 1.1). Another industrial survey of the past decade shows that only one-third of industrial controllers provide acceptable performance (Srinivasan and Rengaswamy, 2005).

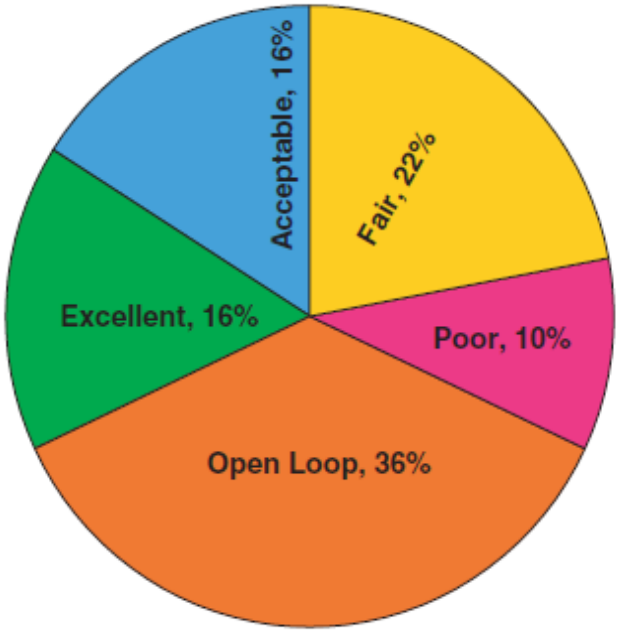


**Figure 1.1:** Performance of industrial controllers as studied by Desborough and Miller (2002).

Being the safeguard of process plant and pipeline industries, the performance of these control loops is critical when the chemical process plants and pipelines involve hazardous materials, concentrated acidic and basic solutions, high temperature and pressure operations and so on. But the control loops rarely give the 'fault-tolerant' performance (Raghavan, 2004), though the goal of designing these controllers is to achieve robust plant performance against undesired conditions. Too many false alarms make the operator reluctant about taking actions when there is a real alarm. Thus, faults in process control and monitoring system can cause large-scale accidents where processes suffer large financial losses due to disastrous impact such as loss of equipment, significant harm to the environment and human life (Izadi et al., 2009a,b, 2010).

It is therefore, fault detection, diagnosis and remediation methods for both unit operations and control instruments are of great importance. The focus of this Ph.D. research is to develop a broader set of solutions for two such problems. Identifying normal process transients such as start-up, shutdown or operating condition changes to get the best result

from the process monitoring system is another goal of this thesis.

## 1.2 Types of Fault

According to their appearance in the system equations, faults can be broadly classified as both additive and multiplicative (Baillieul and Samad, 2015; Raghavan, 2004). Additive faults are mainly due to the sensor or actuator biases or shifts. Other additive process faults such as leaks in the plant or pipeline also fall in this category (Figure 1.2). On the other hand, when the fault is with the system model such as a change in the model parameter, or even the change in the model structure are categorized as multiplicative faults. Figure 1.2 gives an i

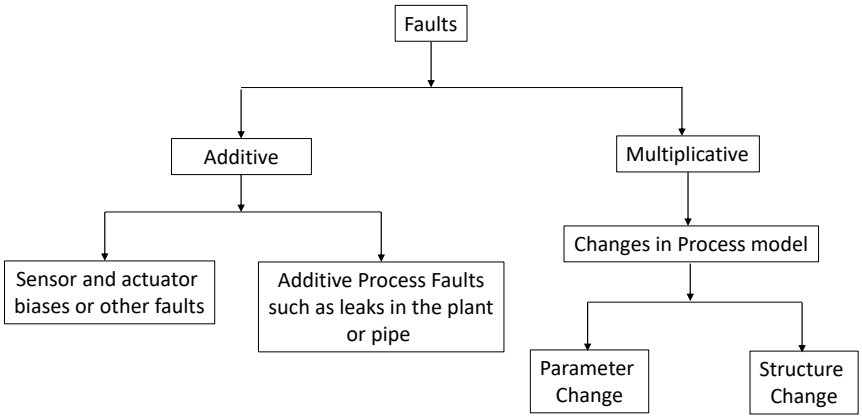


**Figure 1.2:** Classification of faults.

## 1.3 Basic Steps of Fault Detection Algorithm

Highly interconnected process plants consist of sophisticated equipment and control instruments. Consequently, a fault in one of the unit operations may propagate to other connected units and can affect the efficiency of the entire process plant significantly. Such faults can also be the causes of unwanted accidents which can often lead to severe damage to process units along with environmental implications and possible loss of life.

Fault detection techniques try to detect faults in a predictive mode or as soon as the deviation occurs. Then the root cause of the fault is identified followed by steps to remediate the process to its normal state by some corrective actions (Venkatasubramanian et al., 2003c). The basic steps of fault detection and diagnosis (FDD) algorithm are depicted in Figure 1.3.

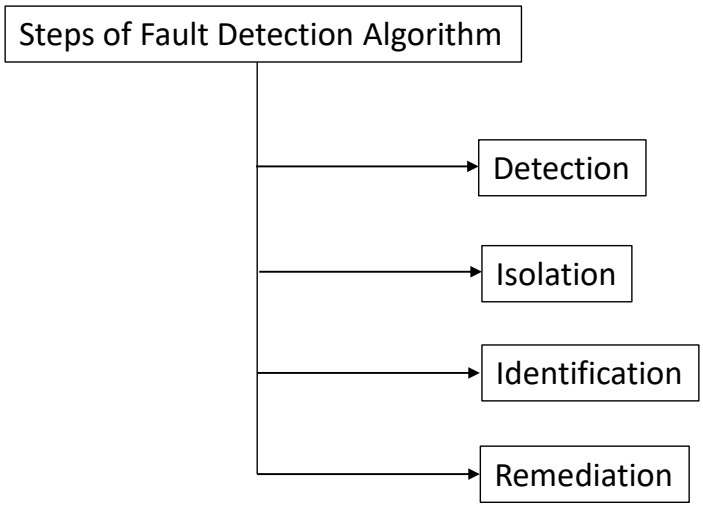


**Figure 1.3:** Steps of fault detection and diagnosis algorithm.

**Detection** The first step of a fault detection algorithm is to indicate whether there is a fault in the process or not. This step is very crucial because if a real fault is missed (missed alarm), a disastrous accident may occur. On the other hand, if it is a false alarm when there is no fault (false-positive), the process may suffer financial loss due to unnecessary shutdown.

**Isolation** After a fault has been detected, the second step of an FDD algorithm is to find out the root cause of the fault or pinpoint the fault location. As there are thousands of control loops and other equipment and unit operations in a process, if the FDD algorithm fails to detect the exact cause or location of the fault, it may take a longer time to find the root cause or location manually. The process may need to be shutdown during this manual search causing financial loss that grows with time. Generally, this step is the most difficult part of an FDD algorithm (Raghavan, 2004).

**Identification** Once the fault is detected and root cause or location of the fault is determined, the next step is to find out the size of the fault. If the size of fault can be estimated exactly, the remediation or reconfiguration step becomes easier since, at this point, the user has all the necessary information of the fault (Raghavan, 2004).

**Remediation** In this final step, the process may need to go through some modification of the control logic (Hwang et al., 2010) or other changes to nullify the root cause of the fault. Otherwise, a similar fault may arise again in future. Because the presence of a fault

indicates that the current configuration of the process is not optimal or robust. The success of this step depends upon the expertise of the operator. Generally, the reconfiguration suggestions are checked by process experts before starting up the plant/process after the event of a fault.

## 1.4 Existing FDD Methods

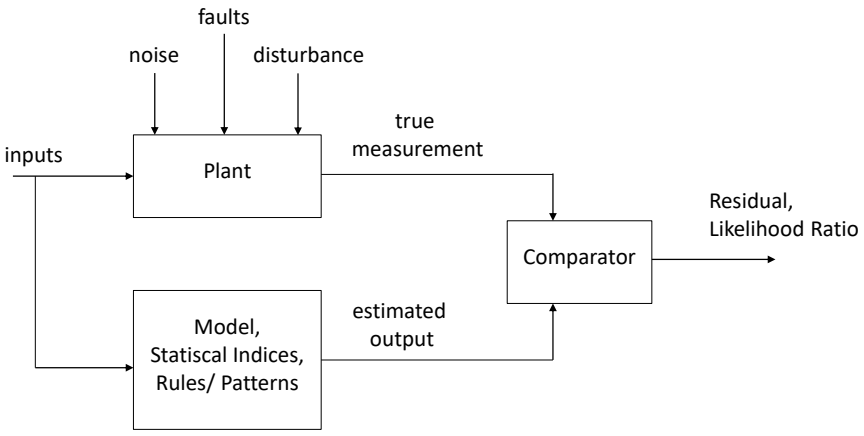


**Figure 1.4:** General schemes of FDD.

The basic idea of FDD algorithm is to compare the real plant measurements with - preset limits with or without having any knowledge of the plant model (Figure 1.4) (Baillieul and Samad, 2015). Hence, fault detection techniques can be divided into three broader classes a) model-based methods, when the real plant measurement is compared with the estimates obtained from the explicit plant model (Venkatasubramanian et al., 2003c,a), and b) data-driven methods or statistical methods where plant data are used to derive statistical models and then the estimates from these statistical models are compared with the real measurements (Venkatasubramanian et al., 2003b; Baillieul and Samad, 2015). c) where neither the explicit plant model nor the statistical model from data is used, rather heuristics knowledge or causal analysis are used to set the limits of the comparison (Venkatasubramanian et al., 2003a; Raghavan, 2004). None of these methods are completely self-sufficient and they typically require complements or combinations of each other (Venkatasubramanian et al., 2003a,b). Figure 1.5 shows the detailed classification of existing FDD method.

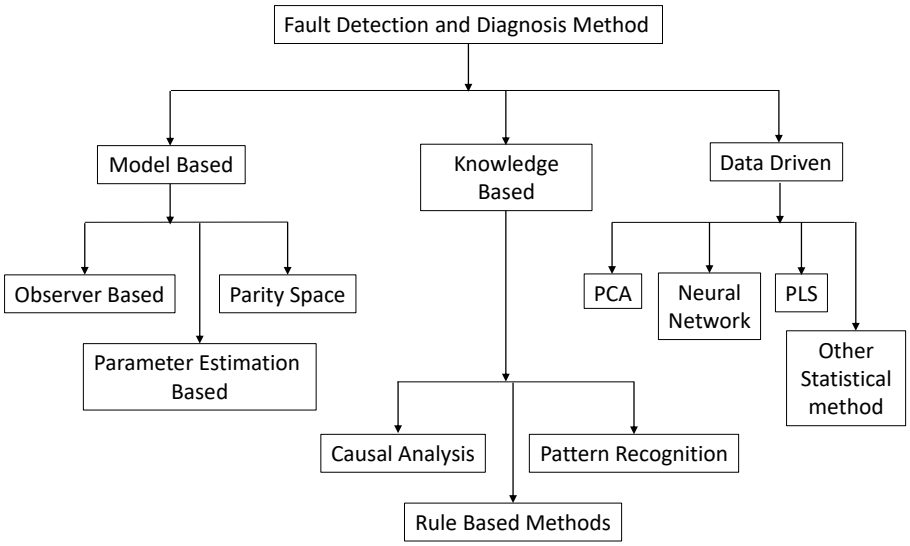


**Figure 1.5:** Classification of fault detection and diagnosis algorithm.

### 1.4.1 Model Based Methods

The model-based methods consist of four separate stages. According to the differences in these stages, the methods differ from one another. The stages are :

**Modeling the System** The system model is derived from the dynamic models of the system or by their simplified version. The system model as shown in equations 1.1 may contain uncertainties (such as $\Delta A, \Delta B, \Delta C, \Delta D$) in parameter and it incorporates the process and measurement noise $(v_{k-1}, n_k)$ as well (Hwang et al., 2010). Here, $x_k$ is the state vector, $u_k$ is the system input which is known and $z_k$ is the measurement vector.

$$\begin{aligned} x_k &= (A + \Delta A)x_{k-1} + (B + \Delta B)u_{k-1} + E_1 v_{k-1} \\ z_k &= (C + \Delta C)x_k + (D + \Delta D)u_k + E_2 n_k \end{aligned} \tag{1.1}$$

**Modeling the Fault** The faults can be modeled as additive such as $F^a$ (actuator fault) or $F^s$ (sensor fault) in equation 1.2. The multiplicative faults remain in multiplicative form with the state vector $(x_k)$ and system input $(u_k)$ such as $\Delta A_m, \Delta B_m, \Delta C_m, \Delta D_m$ in equation 1.2.

$$\begin{aligned} x_k &= (A + \Delta A + \Delta A_m)x_{k-1} + (B + \Delta B + \Delta B_m)u_{k-1} + E_1 v_{k-1} + F^a_{k-1} \\ z_k &= (C + \Delta C + \Delta C_m)x_k + (D + \Delta D + \Delta D_m)u_k + E_2 n_k + F^s_k \end{aligned} \tag{1.2}$$

**Generation of Residual** The residual is the difference between the true measurement ($z_k^t$) and estimated measurement ($z_k$) from the system model as given by equations 1.1 and 1.2. In mathematical form, the residual can be defined as shown in equation 1.3.

$$r_k = z_k^t - z_k \tag{1.3}$$

Since in a real application, this residual ($r_k$) is corrupted by the process and the measurement noises, unknown disturbances and uncertainties in the system model, the FDD schemes try to generate robust residuals so that the residual will be insensitive to these noises and uncertainties but sensitive to the fault(s). Several methods have been proposed for this purpose (Hwang et al., 2010) as listed below -

- observer or state estimator based methods (Basseville et al., 1993)
- parity relations methods (Baillieul and Samad, 2015)
- optimization based methods (Hwang et al., 2010)
- parameter estimation method (Isermann, 2005; Venkatasubramanian et al., 2003c)
- stochastic approach (Venkatasubramanian et al., 2003c; Hwang et al., 2010)
- system identification approach (Basseville et al., 1993)

For isolation of the fault, the residuals generated should be able to discriminate between different types of faults. This can be done in two ways (Hwang et al., 2010; Venkatasubramanian et al., 2003c).

- In the directional residual approach, the residual vector is generated in such a way that it lies in a specified direction in the residual subspace corresponding to each type of fault. By determining the direction of the residual, the type of fault is identified.
- In the structured residual approach, each residual vector is designed to be sensitive to a single or selective set of faults, and insensitive to the rest.

**Decision Making** Once the residual is generated by maintaining all the required criteria as given by above three stages, the next stage is to determine whether there is a fault or not. If there is a fault, the location and type of the fault need to be identified based on some statistical tests of the residuals (Hwang et al., 2010). The commonly used statistical tools are -

- sequential probability ratio test (SPRT) (Wald, 1973)

- cumulative sum (CUSUM) algorithm (Basseville et al., 1993)
- generalized likelihood ratio (GLR) test (Basseville et al., 1993)
- local approach (Basseville et al., 1993)

However, the model-based methods face some practical limitations in real-time fault detection application. Firstly, development of a first principle model is difficult. In most cases, uncertainty of the model parameter makes it difficult to match with the real scenario. If the system identification or black-box modeling approach is used to derive the model, those models often require open loop test which might be a practical limitation. Moreover, the black-box models often have very limited application compared to the first principle model. Secondly, computational complexity associated with models are very high (Venkatasubramanian et al., 2003c). Thirdly, most of the process models are non-linear which is an added complexity.

### 1.4.2 Statistical or Data Driven Methods

To reduce the complexity of deriving the model as well as the computational cost, data-driven FDD algorithms can be used (Venkatasubramanian et al., 2003b). The basic idea of statistical based methods is to infer some statistical indices for normal condition data such as the mean and variance of Gaussian density function as obtained from time-series data during normal operating condition, principal components and their loading vectors in the principal component analysis (PCA) associated with normally conditioned data (Baillieul and Samad, 2015) or the latent variables of partial least square (PLS) method. Since, PCA is a time-invariant method, improved variants of PCA and PLS's are used in a field of fault detection (Venkatasubramanian et al., 2003b). When there is a fault in the process, these statistical indices will show a discrepancy from those in the normal condition behavior. Thus, the fault is detected using the decision making techniques discussed above (Venkatasubramanian et al., 2003b). However, though the statistical tools mentioned above can handle high dimensional and highly correlated process-variables, these methods cannot differentiate between different types of faults as they do not possess information about each type of fault specifically (Venkatasubramanian et al., 2003b).

Another popular statistical method is the classification algorithm where different types of faults and normal condition are considered as classes. The true measurements are then divided into these classes according to some statistical distance (such as Euclidean or Manhattan) or classification rule (such as Bayesian, quadratic or piecewise classifier). If the class of any specific fault is empty, then the process is free from that fault. However, the

efficiency of the classification algorithm depends upon the proper choice of the classifier for a specific system. Improper choice of classification rule may result in false or missed alarm when there is significant overlapping between classes (Venkatasubramanian et al., 2003b).

Classification is a supervised statistical problem, whereas clustering is an unsupervised problem. In case of clustering, the algorithm itself decides how many classes will be there by statistically analyzing the data based on some suitable distance or similarity matrix. Secondly, it requires a representative pattern against which the similarity of other patterns can be checked (Venkatasubramanian et al., 2003b). A recent popular method for detecting fault is the neural network-based method and its variants. The key idea of this method is the feature or pattern extraction from a large set of training data (Nielsen, 2015). This knowledge is then used to identify whether the system is showing normal behavior, or it has some faults (Venkatasubramanian et al., 2003b). However, the neural network methods (NN) require large data-sets, otherwise the detection can mislead the operator.

### 1.4.3 Knowledge Based Methods

The knowledge-based FDD techniques uses some specific knowledge of the process or fault and logical approaches to detect the fault rather than the detailed model of the system or applying extensive statistical tools on a large set of data. Sometimes the basic relationship between the fault and symptoms are partially known. Hence using the causal relationship (fault $\rightarrow$ event $\rightarrow$ symptom), the fault can be detected (Isermann, 2005). Sometimes, the pattern of a specific fault is known as a prior. Using pattern recognition techniques (Venkatasubramanian et al., 2003b) these faults can be detected. However, these knowledge-based methods can only be applied in special cases where specific knowledge, rule or pattern is available. In some cases, a combination of all the three types of FDD method is used to get better results (Venkatasubramanian et al., 2003b).

## 1.5 Structure of the Thesis

Faults are special types of changes which have negative impacts on pipeline and processes. In this thesis, two major industrial faults were studied, and novel detection and remediation methods were developed along with a novel change detection algorithm. Chapter 2 of this thesis introduces the Kantorovich distance (KD) based change detection algorithm with application to the Tennessee Eastman Process. The Kantorovich distance is the minimum distance required to transfer all the elements of a distribution to a second distribution. The first fault addressed in this thesis is the pipeline leak detection. Pipeline is one the safest

ways to transfer oil and gas products to consuming destinations rather than using truck, train or ship. But these pipelines may suffer from leaks which can be the cause of huge financial and environmental losses. In chapter 3, the pipeline hydraulic models are discussed along with modeling of a leak and leak signature.

A novel model-based pipeline leak detection algorithm is proposed in chapter 4. A Monte Carlo simulation based state estimator, the Particle Filter (PF), was used to estimate the unknown states and by comparing the residual between the estimated states and the available measurement with a leak signature, the leak was detected. KD based change detection algorithm was used to localize the leak. This algorithm was applied to simulated pipeline leak(s). However, since particle filter is computationally expensive, and models are corrupted with uncertainty in parameters, this algorithm was not successful in the application to the real pipeline.

Chapter 5 consists of a novel leak detection algorithm using the Kantorovich distance (KD) where the concept of change detection was applied to leak detection problem. Since the change detection algorithm is completely data-driven, the problems corresponding to model-based LDS is resolved. This algorithm was applied to both simulated and real-time test leaks during the annual regulatory tests of Suncor pipelines. The algorithm successfully detected both a small leak (1% of nominal flow-rate) and a large leak (7% of nominal flow-rate). The detection time was much less than that of the existing commercial leak detection and localization system (LDS). The leak localization efficiency is also comparable to the existing LDS.

The second fault is the control valve stiction problem which is a common source of oscillations in process industries and pipelines. Chapter 6 of the thesis gives a formal definition of stiction along with the valve mechanism. A novel data-driven stiction detection method is proposed using the KD algorithm. Then a novel stiction compensation method is also proposed where the compensating signal is proportional to the error between the process-variable (PV) and desired setpoint (SP). The proposed method was applied both on a simulated system and also evaluated on an experimental setup with a sticky of valve situated in the 'computer process control (CPC)' laboratory at the University of Alberta. An improved version of the compensation algorithm is proposed in Chapter 7, in which the compensating signal is ramp modulated to achieve the best performance in terms of valve travel (VT) and integrated absolute error (IAE). This improved version of stiction compensation algorithm was applied successfully to a pilot plant equipped with a sticky valve with built-in valve positioner.

Finally, Chapter 8 gives concluding remarks for the thesis including a summary of the main contributions of this thesis along with recommendations for future research directions.

# Chapter 2

# Change Detection Using Kantorovich Distance Algorithm

Most of the existing process control and monitoring systems analyze steady-state plant operational data to establish the alarm thresholds. However, process operational and product grade changes may result in transients. These transients will result in false alarms, which misinform the operator that the facility is operating outside its 'normal' operating zone. Significant transient conditions will result in alarm-floods which have the potential to overwhelm the process operators. In this chapter, a novel change detection algorithm has been proposed, based on the Kantorovich distance, which will assist the operator in detecting dynamic changes in the process plant. Incorporating the result of the change detection algorithm with the existing process monitoring tools may reduce the amount of false alarms as well as missed alarms. The proposed change detection method was applied successfully to a simulated pipeline, as well as on historical pipeline data. The algorithm has also been successfully applied to real pipeline data and benchmark Tennessee Eastman (TE) process in online (real-time) mode.

## 2.1 Literature Review

Pipelines and chemical processes go through many transient conditions with the most extreme being start-ups and shutdowns. During these transitions, many process monitoring and alarm algorithm systems fail and often cause false alarms and alarm-floods (in the worst case (Adnan et al., 2011). This is because the thresholds of most of the process monitoring variables are preset and they are unaware of the transitions taking place (Zhu et al., 2014). So, if there is a supporting algorithm which can detect the change-point during the transitions and the results from this change detection algorithm can be coupled with the current process monitoring systems, the reliability of process plant monitoring will increase.

This study focuses on this specific target of developing a reliable online change detection algorithm which can detect process transients efficiently.

In process industries, the idea of fault detection and process monitoring is widely used to detect abnormal situations. Faults are special types of changes which represents the degraded condition of the process. All the changes are not necessarily faults. Change in product throughput, operating condition, product specification or grade change are common examples where the process is running smoothly and the setpoints of concerning variables have been changed by the operator purposefully to meet the process requirements. But even then, since the preset thresholds of the process monitoring variables have no idea what is coming up (Zhu et al., 2014), false alarms may be raised and in extreme cases it may also cause alarm-floods (EEMUA, 2007; ISA, 2009). Similar scenarios happen when shutting down a plant for planned maintenance and starting it up again after the maintenance. These are not faults but purposeful or pre-scheduled changes. To avoid alarm-floods in such cases, most of the time the existing process monitoring systems are either kept turned off or the alarm thresholds are relaxed. So, if a fault occurs during these transient periods, there is a high chance that the fault will not be detected by the available systems. On the other hand, faults are unwanted and unplanned changes. So, in summary, it can be said that fault detection and diagnosis (FDD) and process monitoring algorithms are specific types of the broader class of change detection algorithms.

Process monitoring and fault detection systems can be broadly classified into two classes - a) model-based and b) data-driven (Adnan et al., 2011; Izadi et al., 2009a). The model-based methods for process monitoring and fault detection mainly works on the residual between a process model estimation and real measurements (Venkatasubramanian et al., 2003d). Some common model-based process monitoring systems are residual evaluation, parity relations, optimal Kalman filtering, parameter estimation by using parametric model (Venkatasubramanian et al., 2003d). Model predictive control (MPC) algorithm is one the best methods for process monitoring and control which also uses process models (van den Boom and Backx, 2010). The inherent problem of all model-based methods is identifying an accurate model for the real system. No model can imitate the real system perfectly. Besides, solving the model also takes more time in comparison with their data-driven counterparts, which might be a practical limitation for many real applications.

Among the data-driven methods various multivariate statistical tools such as CUSUM (cumulative summation), PCA (Principal Component Analysis), MPCA (Multiway Principal Component Analysis), PLS (Partial Least Squares), PCR (Principal Component Regression), MLR (Multiple Linear Regression), ILS (Inverse Least Squares), EFA (Evolving Factor Analysis), MCR (Multivariate Curve Resolution) (Basseville et al., 1993; Izadi et al.,

2009a; Wise and Gallagher, 1996) are often used. Data-driven models can also be derived for process monitoring using system identification or statistical models using PCA, PLS and time-series models (Izadi et al., 2009a).

Other popular statistical algorithms in the field of change detection are likelihood ratio test based (Kawahara and Sugiyama, 2012; Liu et al., 2013) methods. Different variants of likelihood ratio such as the generalized likelihood ratio (GLR) (Basseville and Benveniste, 1983; Chetouani, 2012), maximum likelihood estimation (MLE) have also been studied. Chetouani (2012) proposed a change detection algorithm combining with the GLR and artificial neural network (ANN) methods. To handle the nonlinear behavior of distillation column, the neural model was derived using non-linear auto-regressive with exogenous input (NARX) algorithm applying the experimental design algorithm.

Another type of change detection method is subspace method where a sub-space of the given time series data is identified using PCA, ICA (independent component analysis) or other subspace identification algorithm (Kawahara et al., 2007; Kawahara and Sugiyama, 2012). In these methods, change is identified from trajectories in past and present intervals. The distance between the subspaces are the measures of difference (Liu et al., 2013). Liu et al. (2013) described a change detection using relative density based methods. This method is non-parametric based but works offline.

Most of the methods discussed so far are mainly for offline detection. Author of Adams and MacKay (2007) has described a Bayesian method to detect online change which assumes i.i.d. (independent and identically distributed) data and exponential family likelihood. Both assumptions may not be true for real data. Moreover, no example with online detection is shown. Another Bayesian algorithm based dynamic alarm management strategy for the chemical process in combination with AISFD (Artificial Immune System-based Fault Detection) is described in Zhu et al. (2014). In this case, the noise was assumed as i.i.d. Gaussian distribution and the conditional mean was assumed as an unknown linear function. Moreover, quite a bit of historical data is required to generate the so-called antibody of AIS. Online learning or kernel-based methods are another orientation of change detection research. These methods are used to extract feature space to solve problems like classification, regression or novelty detection (Kivinen et al., 2004). This method requires a large amount of past data as training data-set and still in some experimental cases the method showed inaccurate detection. Moreover, calculation of the loss-function may be tricky.

Kawahara and Sugiyama (2012) proposed direct density ratio based non-parametric online method which can be applied online according to the authors. To estimate the density ratio the authors assumed Gaussian kernel model which ultimately forms a non-linear con-

vex optimization problem which may be time-consuming while applying to a real process for online detection. Moreover, this algorithm has too many parameters to be tuned.

In this chapter, a novel optimal-distance (Kantorovich) based change detection algorithm is proposed to detect change points associated with dynamic processes. The novelty of the proposed method is that it is a simple algorithm which ensures optimal solution by solving the linear programming problem. So, the method is faster than the method described in Kawahara and Sugiyama (2012) and it ensures optimal solution since no estimation is required for the density ratio. The i.i.d. assumption is also not needed for the proposed method. It has a fewer number of parameters to be tuned than the method described in Kawahara and Sugiyama (2012). To initiate the proposed method, only a few past samples are required unlike other data-driven methods. Only a few of the above-discussed change detection methods are applied to process industries till today. In this chapter, case studies with benchmark Tennessee Eastman process in online as well as offline change detection for both simulated and real data are also described. Incorporation of this change detection algorithm with the existing process monitoring system will be addressed in future studies.

This chapter consists of six sections. Section 1 provides the scope of the proposed change detection algorithm and a short literature review of available process monitoring and change detection algorithms. Section 2 defines the Kantorovich distance (KD), the main tool of the proposed change detection algorithm. It also describes the computational implementation of KD along with numerical examples. Section 3 describes the application of KD-based change detection algorithm on few case studies in offline mode. Three general simulated case studies, one simulated pipeline flow problem and one evaluation of real industrial data are included. Section 4 consists of the online implementation case studies and the challenges and remedies for online application of KD. Verification of the proposed method was done by detecting simulated and real changes from another set of real data in the online application. Section 5 shows the application of the proposed method on benchmark Tennessee Eastman process in real-time. The chapter ends with concluding remarks and scope for future work in section 6.

## 2.2 The Kantorovich Distance

The Kantorovich distance (KD) is defined as the minimum distance (transportation cost) required to trans-locate mass from one probability distribution to another probability distribution. This concept was first introduced to handle different organizing and planning problems to either maximize the profit or minimize the cost solving the linear programming for-

# Definition: Kantorovich Distance (KD)

## The minimum distance travelled to transfer all the elements of probabilistic distribution to another probabilistic distribution

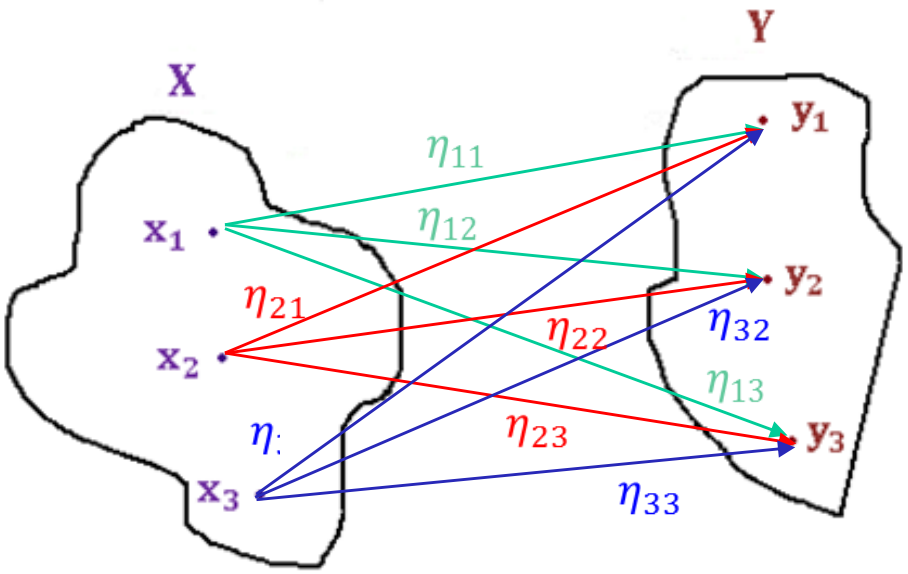


**Figure 2.1:** Illustration of transportation problem.

The basic idea of KD is explained in Figure 2.1. While transferring all the elements of distribution $X$ to the new distribution $Y$, the minimum "transportation" distance required to complete this transfer process is the Kantorovich distance. The more dissimilar the distributions are, the larger will be the associated minimum cost and hence this distance gives an idea of the degree of dissimilarity between the two distributions. For simplicity, it was assumed that in distribution $X$ (original distribution) there are only three elements which are to be transferred to distribution $Y$ (new distribution). There are three different paths ($\eta_{11}$, $\eta_{12}$ and $\eta_{13}$) to transfer the component $x_1$ to the available new positions of distribution $Y$ such as $y_1$, $y_2$ and $y_3$. These are denoted by green colors in Figure 2.1). Similarly, there are three different paths for each of $x_2$ and $x_3$ to complete the transfer and these paths are denoted by red and blue colors respectively in Figure 2.1.



The minimum transportation distance (KD) from distribution $X$ to distribution $Y$ can be

mathematically expressed as

$$D_K = \min_{\eta_{ij} \geq 0} \sum_{i=1}^{N} \sum_{j=1}^{N} d(x_i, y_j)\eta_{ij}$$
$$\text{s.t.} \sum_{i=1}^{N} \eta_{ij} = b_j$$
$$\sum_{j=1}^{N} \eta_{ij} = a_i$$

where

- $D_K$ = Kantorovich distance
- $d(x_i, y_j) = \sum_{\tau=1}^{t} |x_i^\tau - y_j^\tau|$, a measure of the dissimilarity between element $x_i$ and $y_j$
- $a_i$ = original probability of an element of $X$ and $\sum_{i=1}^{N} a_i = 1$
- $b_j$ = original probability of an element of $Y$ and $\sum_{j=1}^{N} b_j = 1$
- $\eta_{ij}$ = decision variables that represent the mass transportation plan.

The KD calculation is only based on the data, so it is not restricted by the distribution of the data itself. The application of Kantorovich distance is not also limited by the independent and identically distributed (i.i.d.) random variable assumption.

### 2.2.1 Computational Implementation

There are two user-defined parameters for the implementation of the Kantorovich distance: a) number of segments ($m$) and b) number of samples ($k$) in each segment which have been shown pictorially in Figure 2.2. In this figure distribution $X$ has $m$ segments $x_1$, $x_2$,..., $x_m$. Each of these segments has $k$ samples such as segment $x_1$ has elements as $x_{11}$, $x_{12}$..., $x_{1k}$. Similarly, distribution $y$ has $m$ segments $y_1$, $y_2$,..., $y_m$ where each of these segments have $k$ data points such as $y_{11}$, $y_{12}$..., $y_{1k}$ are the data points of segment $y_1$ (see Figure 2.2). Each segment of the first distribution ($X$) is compared with each of the segments of the second distribution ($Y$). Thus, distribution $X$ is comprised of samples 1 to $(m + k - 1)$ and distribution $Y$ is comprised of samples $(m + 1)$ to $(2m + k - 1)$ each having total $(m + k - 1)$ samples.

The distance ($d$) used in this study is the absolute distance (norm 1) between the elements of segment $x_1$ with the corresponding elements of $y_1, y_2, ..., y_m$ and so on. For example, second element of segment $x_1$ ($x_{12}$) will always be compared with the second elements

of $y_1, y_2, ..., y_m$ that means with $y_{12}$, $y_{22}$ to up to $y_{m2}$. Once comparison of all the elements of $x_1$ is done with the corresponding elements of $y_1, y_2, ..., y_m$, then each element of $x_2$ is compared with the corresponding elements of $y_1, y_2, ..., y_m$ and so on. The absolute distances for these comparisons are recorded and used to determine whether there is significant change between the two distributions $X$ and $Y$.

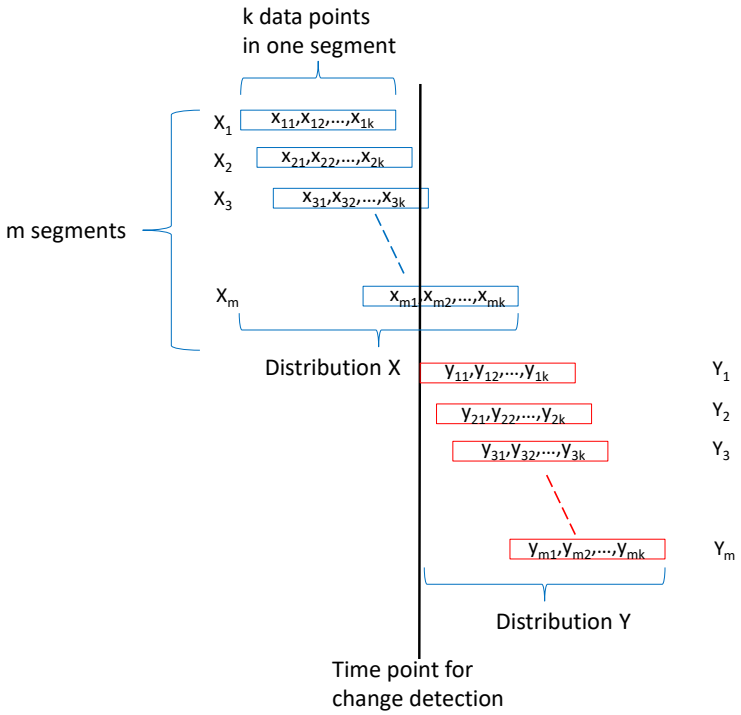


**Figure 2.2:** Data segmentation for Kantorovich distance calculation.

As an example, if the data is of total 8 samples such as

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix}$$

then to calculate the KD at $t = 4$, the distribution $X$ and $Y$ will be as follows with $m = 3$ and $k = 2$:

$$X = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 4 \end{bmatrix}, \quad Y = \begin{bmatrix} 4 & 5 \\ 5 & 6 \\ 6 & 7 \end{bmatrix}$$

So, to calculate KD for the first time $2m + k - 1$ samples are required. In case of offline detection all the samples are readily available but for online detection the algorithm needs to wait till $2m+k-1$ samples are accumulated which is the source of detection delay in this method. The corresponding distance matrix for this simple example will be the distance

calculated between segments in distributions $X$ and $Y$. Now by comparing with Figure 2.1, distribution $X$ has 3 segments as $x_1$, $x_2$ and $x_3$ each of which has 2 samples. Similarly, distribution $Y$ has 3 segments as $y_1$, $y_2$ and $y_3$. So, $x_1$ has 3 ways ($\eta_{11}$, $\eta_{12}$ and $\eta_{13}$ ) to be transferred to distribution $Y$. The distance matrix for this time instant will be:

$$d(x_i, y_j) = \begin{bmatrix} 6 & 8 & 10 \\ 4 & 6 & 8 \\ 2 & 4 & 6 \end{bmatrix}$$

It is noteworthy that there is overlapping between each segment and some overlapping between the two distributions. The distributions are separated by the time point for change detection as shown in Figure 2.2. This is how the optimization problem is set up for KD calculation. The rest is done by solving the linear programming problem.

### 2.2.2 Numerical Example

Two Gaussian distributions were assumed as: $X \sim N(m_1, S_1)$, $Y \sim N(m_2, S_2)$, with $m_1 = 5$, $S_1 = 0.3$, $m_2 = 6$ and $S_2 = 0.1$.

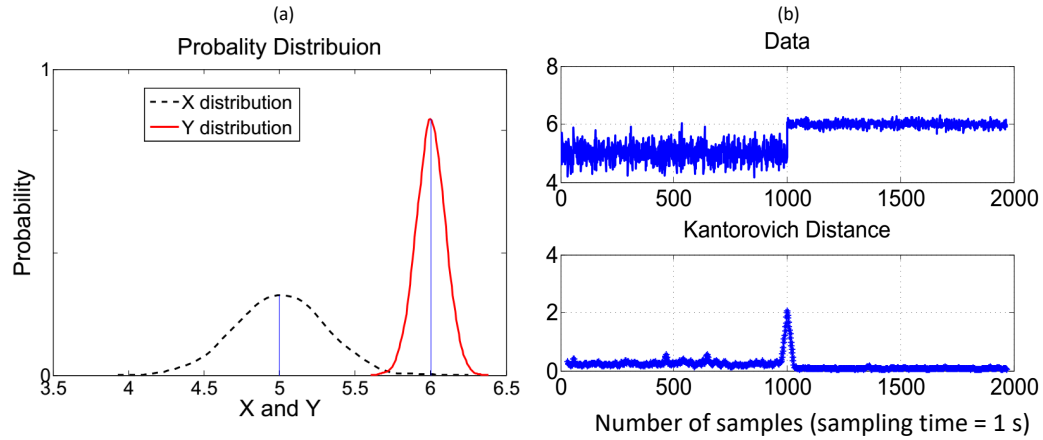


**Figure 2.3:** Panel ($a$) shows the probability distributions, $X$ and $Y$. Upper part of panel ($b$) shows the formation of time series data with these distributions and lower part shows the corresponding KD score.

Let, each of these distributions have 1000 samples. Panel $a$ of Figure 2.3 shows the probability distributions of $X$ and $Y$. The Kantorovich distance is the minimum distance required to transfer all the elements of distribution $X$ to distribution $Y$.

Next, $X$ and $Y$ will be assumed as time series data, where $X$ exists for the first 1000 samples and $Y$ exists for the next 1000 samples. After 1000 samples, the change was introduced in the time series (both in mean and variance). The upper part of panel $b$ of

Figure 2.3 shows this time series data and the bottom part shows the calculated KD for this data. It is clear from this figure that the mean of the data set has been shifted from 5 unit to 6 unit at $1000^{th}$ sample. It is also to be noted that the in the first 1000 samples the data is noisier than the data in the last 1000 samples. This is because the standard deviation of $X$ is higher than the standard deviation of $Y$. The bottom panel shows that the KD algorithm was effective to detect this change. One can see that KD score for the first 1000 samples is higher than the second 1000 samples since the variance of the former was higher than the later. So, changes in both mean and variance can be detected with the proposed method by choosing suitable threshold.

### 2.2.3 Choice of Parameter $m$

**Offline Mode**

Offline change detection is used to analyze the data after a change/fault has already occurred to take the future preventive step. It is a post-processing step. In case of offline change detection, the algorithm has access to all the data from the beginning of the analysis. This is quite different from the online or real-time change detection where the algorithm gets a new data point in each sampling instant. In this sub-section, choice of parameter $m$ will be discussed with the help of simulated data in offline mode. Online detection will be discussed in the following sub-section.

It is obvious that the larger is the window size of the data required to calculate the Kantorovich distance, the larger will be the detection delay. From the discussion of section 2.2.1 it is clear that the window size of data requires for this calculation is dependent on both parameters m and k. For offline detection, the minimum number of samples required to start the KD calculation is $m + 1$.

Panel *a* of Figure 2.4 shows the performance of Kantorovich distance calculated for the data where mean shift occurred at 200 samples from 0 to 1. The noise standard deviation for this data set was fixed at 0.1. Nine different $m$ values were tried (from 2 to 10) with a $k$ value fixed at 2.

To calculate the Kantorovich distance for the first time, $m + 1$ samples needs to be accumulated in offline mode (panel *b* of Figure 2.4). So, it is obvious that the lower is the value of $m$ for a fixed $k$, the lower will be the time to start the KD calculation but it has no effect on offline change detection time which is either 200 or 201 s(see panel *c* of Figure 2.4). So, in case of offline change detection, there is no detection delay which is as per expectation.

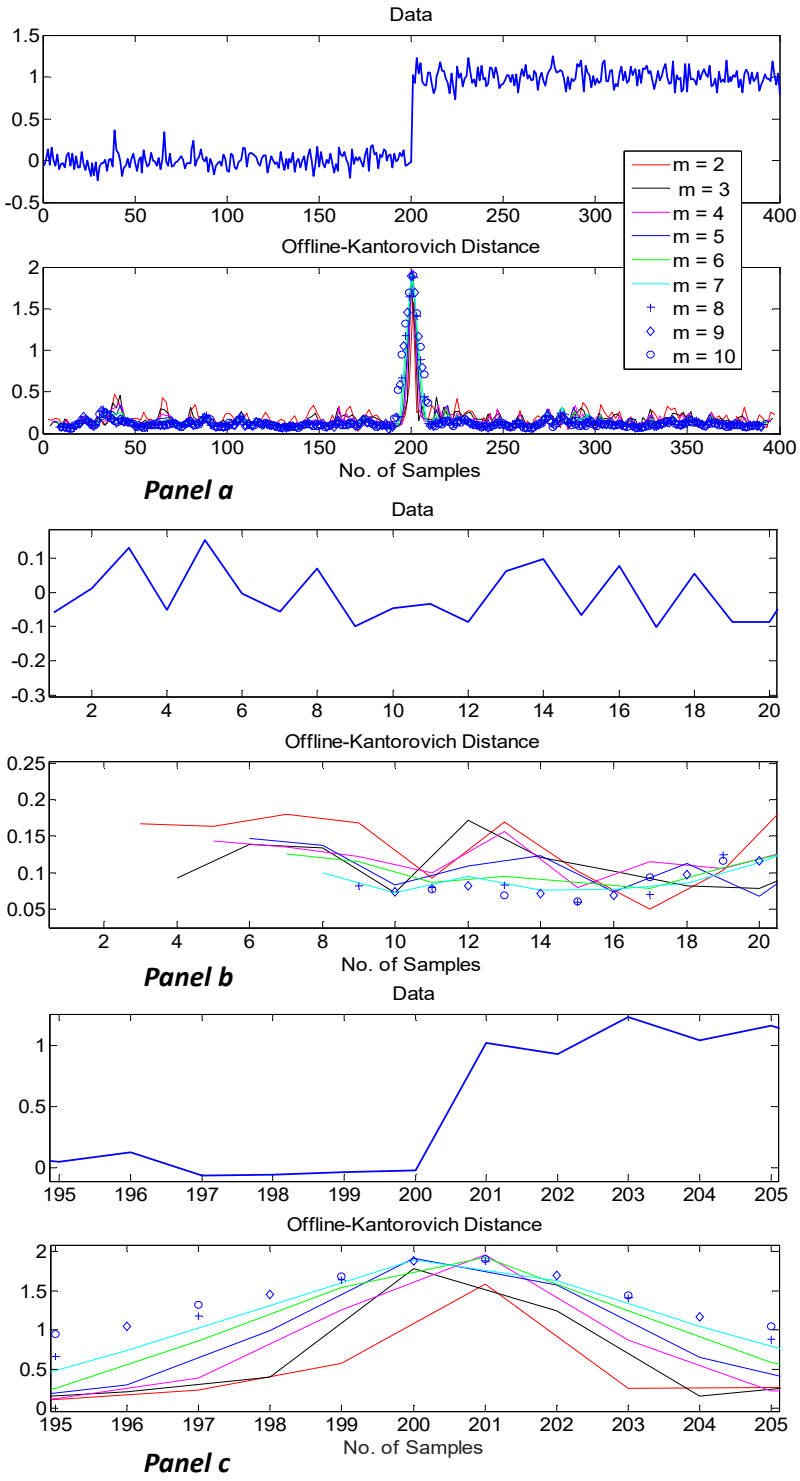


**Figure 2.4:** *a)* Effect of parameter $m$ on **offline** calculation of the Kantorovich Distance, *b)* effect on calculating the KD for the first time, *c)* effect on offline change detection time and smoothness of the KD profile. Panel *b* and *c* are zoomed-in versions of panel *a*.

The smoothness of the Kantorovich distance also needs to be considered. Panel *b* of

Figure 2.4 shows that with the increase of $m$, the Kantorovich distance profile becomes smoother than that of KD associated with lower values of $m$. In this figure, a smaller section of data has been chosen where there is no significant change except the process noise. Still, it is seen that with the value of $m$ around 2 and 3 are giving some spikes due to these noises and average Kantorovich distance (KD) scores are around 0.2 in these cases. But with a value of $m$ at around 9 and 10 has average Kantorovich distance score around at 0.1. The KD profile is smoother for $m$ value 4 and higher comparing to lower values of $m$.

**Online Mode**

In case of online mode, the change detection algorithm gets one new data point (sample) in each sampling instance. So, the algorithm has no idea about the later data points beyond the current and past sampling instances. So, detection gets delayed since the algorithm needs to wait for newer samples which will carry more information about the upcoming change.

In this case, the algorithm waits for $2m + k - 1$ samples to check whether there is an upcoming change or not. The same data-set used for the offline detection in the previous sub-section is used in this case. It is obvious from panel $a$ and $b$ of Figure 2.5 that the larger the value of $m$ is, the larger it will take to start the calculation of KD score in online mode. Panel $b$ of Figure 2.5 is a zoomed-in version of panel $a$ of Figure 2.5. From this panel ($b$), for $m = 2$, the calculation started at sampling instance 5; for $m = 3$, the calculation started at sampling instant 7 and so on. The latest start time of KD calculation is for $m = 10$ at sampling instant 21. Hence the smallest $m$ (that is m =2) will give the smallest time to start the KD calculation.

With the increase in the value $m$, the detection delay will be also increased as seen from panel $c$ of Figure 2.5. This is also a zoomed-in version of panel $a$ of Figure 2.5. It is obvious from this panel ($c$) that the change is detected few samples later than the $200^{th}$ sampling instant. One can see that for $m = 2$, change was detected at $203^{th}$ sampling instant; $m = 3$, change was detected at $204^{th}$ sampling instant and so on. The latest detection time was $210^{th}$ sampling instant for $m = 10$.

The smoothness of the KD profile depends on the $\frac{k}{m}$ ratio. As seen from panel $b$ of Figure 2.5 that with the increase of $m$, $\frac{k}{m}$ decreases and the KD profiles becomes smoother with the decrease of $\frac{k}{m}$ ratio. So, there is a trade-off between smoothness (to reduce false alarm) and detection delay. Hence, it is proposed to use $m = 5$ or greater. In all the subsequent cases, $m = 5$ is used unless otherwise stated.

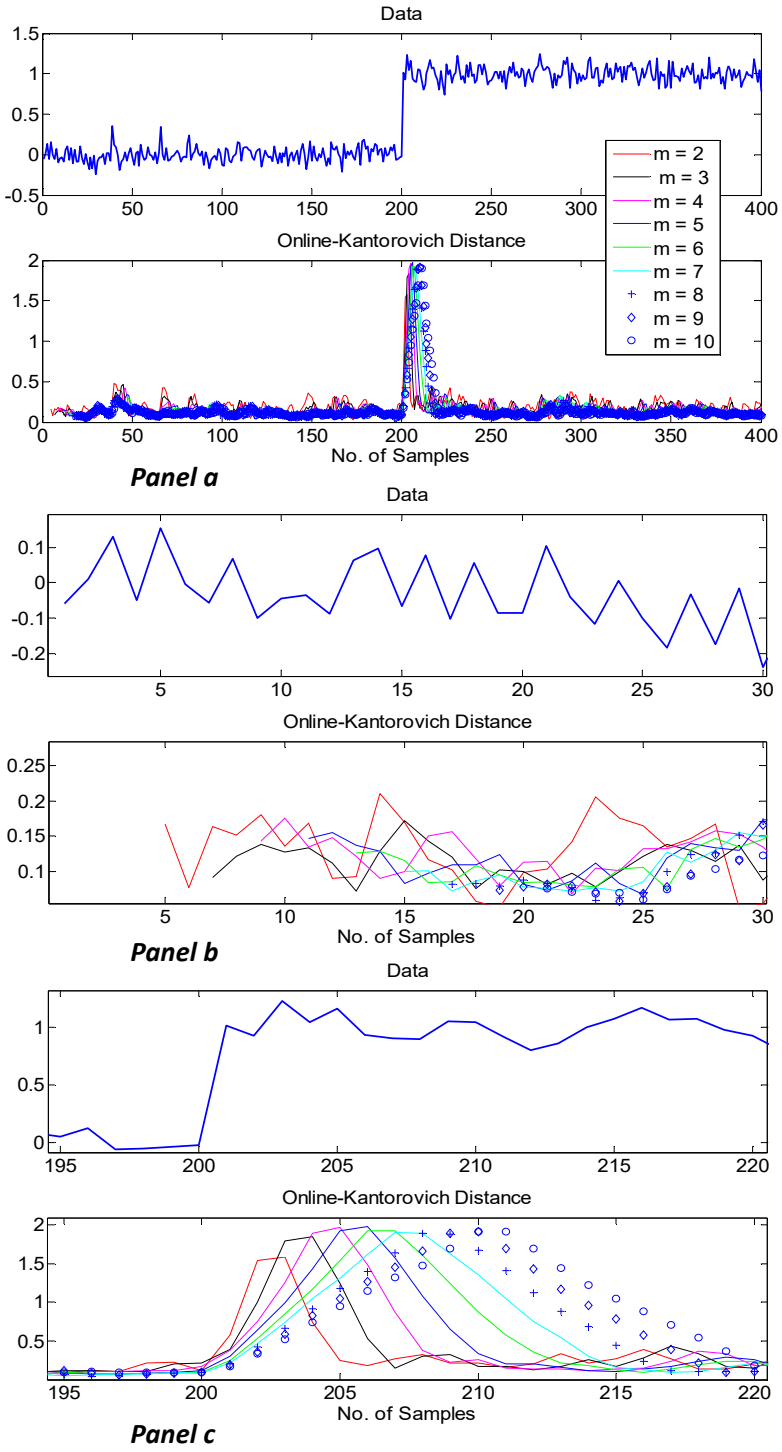


**Figure 2.5:** *a)* Effect of parameter $m$ on **online** calculation of the Kantorovich Distance, *b)* effect on calculating the KD for the first time, *c)* effect on offline change detection time and smoothness of the KD profile. Panel *b* and *c* are zoomed-in versions of panel *a*.

### 2.2.4 Choice of Parameter $k$

#### Offline mode

The number of samples or data-points in each segment of the distribution is represented by the parameter $k$. In this subsection, effect of changing the parameter $k$ is discussed for offline detection with the help of simulated data. Seven different values of k (1 to 7) were varied in this case for a fixed value of parameter $m$. The value of $m$ was fixed at 5 in all these simulations. Panels $a$, $b$ and $c$ of Figure 2.6 show the effect of parameter $k$ on the calculation of KD in offline detection.

From the previous subsection (2.2.3), it is known that for offline change detection the calculation of KD starts at $(m + 1)^{th}$ sample. Hence the start of KD calculation does not depend on the value of $k$ in case of offline change detection. This phenomenon is reflected in panel $b$ of Figure 2.6. From this figure, it is clear that with all the seven different values of parameter $k$, the calculation of KD starts at the same time at $6^{th}$ sample which is the $(m + 1)^{th}$ sample since $m = 5$ in this case.

However, parameter $k$ has an effect on change detection time, the higher the value of $k$, the lowest is the detection time. This phenomenon is prominent from panel $c$ of Figure 2.6, where with $k = 7$ the change is detected at $198^{th}$ sample, whereas with $k = 1$, the detection time is $202^{th}$ sample. The reason behind this phenomenon is that $k$ is the number of samples in each segment. So, higher $k$ means larger number of samples in each segment and when one compares a segment of distribution $X$ with 7 samples in each segment with the segments of distribution $Y$, due to higher amount of overlapping in $X$ and $Y$, later segments of $X$ already have some samples which belongs to $Y$. So, the algorithm gets some hints about the incoming change a bit earlier. This is only possible in case of offline detection.

While calculating the KD score, the absolute distance (norm 1) between the samples of segments of $X$ and $Y$ are summed up for all the samples of a segment. That is why with the increase of samples in a segment (i.e. increase in the value of $k$), KD score also increased which is seen from panel $b$ of Figure 2.6. So, $k = 1$ leads to the lowest KD score and $k = 7$ leads to the highest KD score. Generally, $k$ value near 2 is preferable since it is smoother than higher $k$ values. On the other hand, $k$ value smaller than 2 does not fulfill the purpose of segmentation.

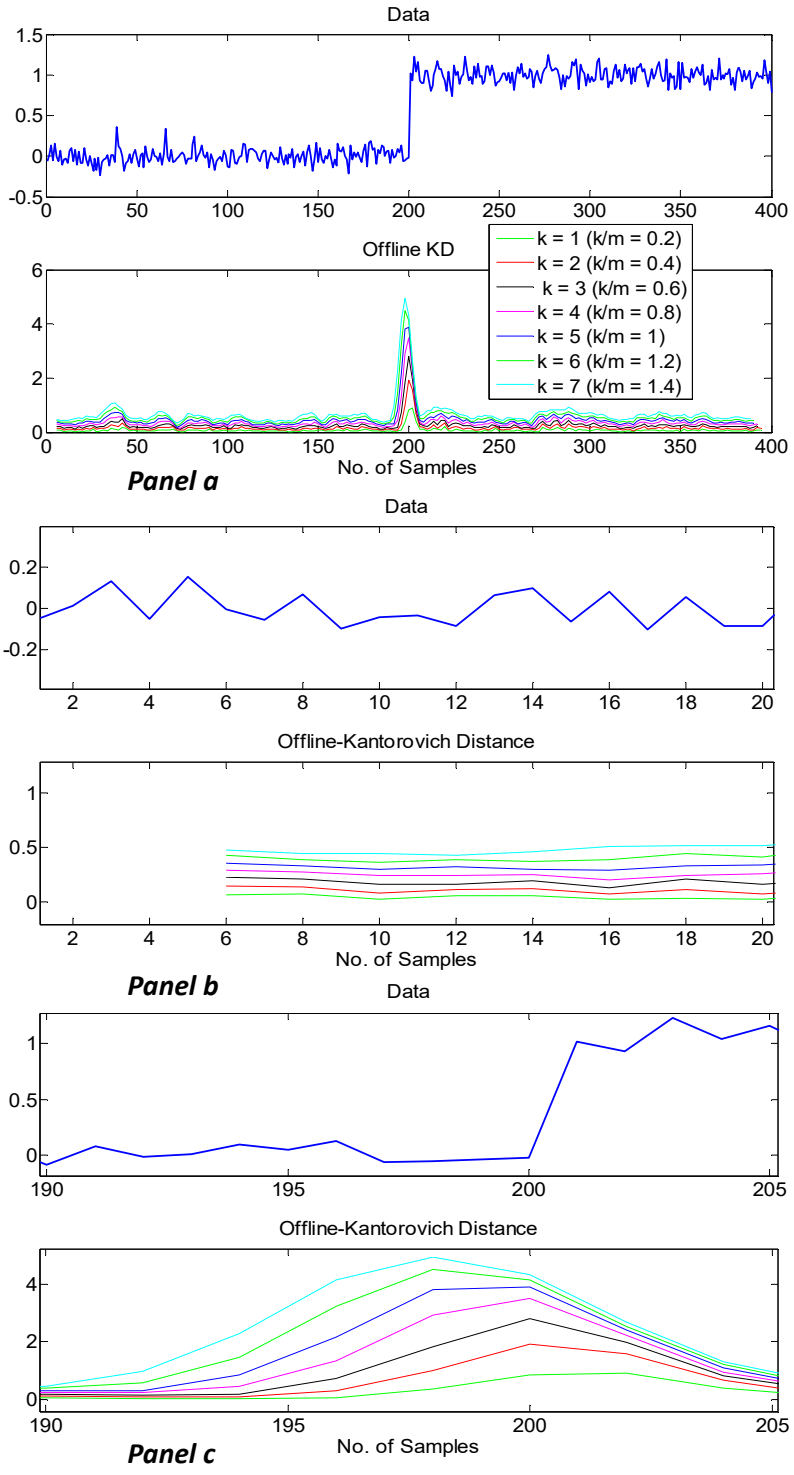


**Figure 2.6:** *a)* Effect of parameter $k$ on **offline** calculation of the Kantorovich Distance, *b)* effect on calculating the KD for the first time, *c)* effect on offline change detection time and smoothness of the KD profile. Panel *b* and *c* are zoomed-in versions of panel *a*.

### Online mode

For online detection, the KD calculation does not start until $2m + k - 1$ number of samples are accumulated. So, in case of online detection, parameter $k$ does have effect on the start of KD calculation which is very clear by panels $a$ and $b$ of Figure 2.7. One can see from Panel $b$ of this figure that for $k = 1$, the calculation of KD started at $10^{th}$ sample since value of $m$ was kept fixed at 5 in all these simulations. Similarly, for $k = 2$, the calculation of KD started at $11^{th}$ sample; for $k = 3$, the calculation of KD started at $12^{th}$ sample; for $k = 4$, the calculation of KD started at $13^{th}$ sample; for $k = 5$, the calculation of KD started at $14^{th}$ sample; for $k = 6$, the calculation of KD started at $15^{th}$ sample and finally for $k = 7$, the calculation of KD started at $16^{th}$ sample.

For the same reason, the detection delay for online mode does increase with the increase of parameter $k$ as can be seen from panel $c$ of Figure 2.7. From this figure, for $k = 1$, the detection time is $205^{th}$ sample whereas for $k = 7$ the detection time is $208^{th}$ sample. Panel $b$ of Figure 2.7 shows that with the increase in parameter $k$, the KD score also increases for online mode as seen in case of the offline mode.

This section gives an idea about how to choose the value of parameter $k$ for the proposed change detection algorithm. From the above discussions, it is clear that lower value of $k$ gives smoother KD score which helps to detect smaller changes. In the rest of this chapter, the $k$ is value kept fixed at 2 unless otherwise stated.

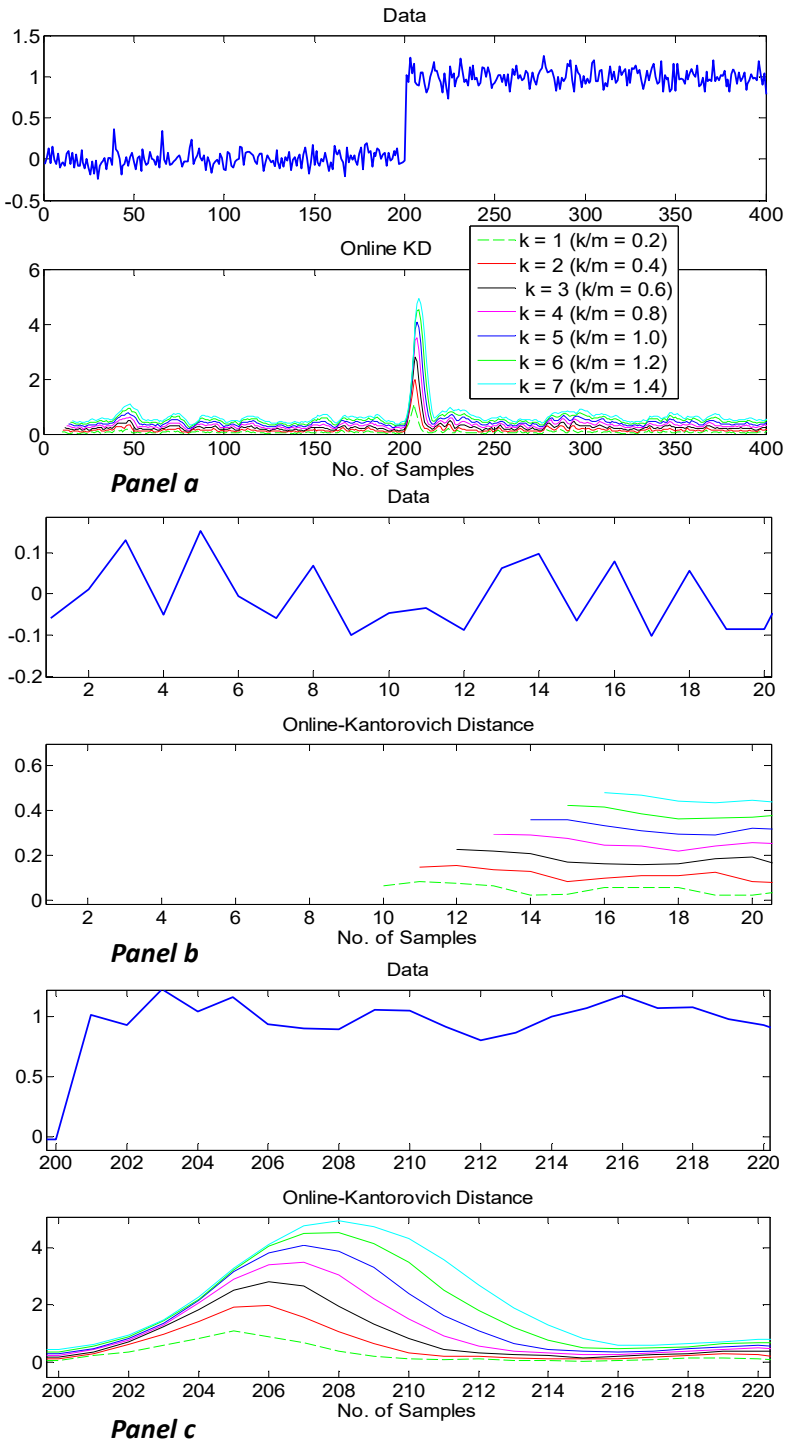


**Figure 2.7:** *a)* Effect of parameter $k$ on **online** calculation of the Kantorovich Distance, *b)* effect on calculating the KD for the first time, *c)* effect on offline change detection time and smoothness of the KD profile. Panel *b* and *c* are zoomed-in versions of panel *a*.

## 2.3 Case Studies with Offline/Batch Change Detection

In case of offline change detection, the change detection algorithm has access to all the data from the very beginning. In this section examples of offline change detection will be discussed. At first, some common types of changes faced in process industries will be discussed with simulated data. Finally, this algorithm was applied to detect changes in both simulated and real pipeline data.

### 2.3.1 Detection of Common Types of Changes in Process Industry

In this section, the performance of the Kantorovich distance based change detection algorithm is studied on some common simulated types of changes in process industries. At first, change in mean and change in variance with simulated data are investigated. Then the performance of KD, on detection of incipient and sharp changes, was analyzed.

**Change in Mean** In many process applications, the word 'change' is used interchangeably with the word 'mean-shift' because process plants often undergo setpoint changes. The setpoint changes in processes refer to shifting the operating condition of the process from one steady state to new steady state which is quite comparable with shifting the mean of the data from an old position to a new position.

Panel *a* of Figure 2.8 shows the efficacy of the proposed Kantorovich distance based change detection algorithm in detecting the mean shift in simulated data. Clearly the mean of the data has been shifted to a higher value at $200^{th}$ sample. It came back to the previous mean value at $400^{th}$ sample. The bottom part of this panel gives the corresponding Kantorovich Distance. There are 2 peaks, the first one is at $200^{th}$ sample and the second one is at $400^{th}$ samples. Clearly, the algorithm detected here to setpoint changes - a) step-up at $200^{th}$ sample and b) step-down at $400^{th}$ sample.

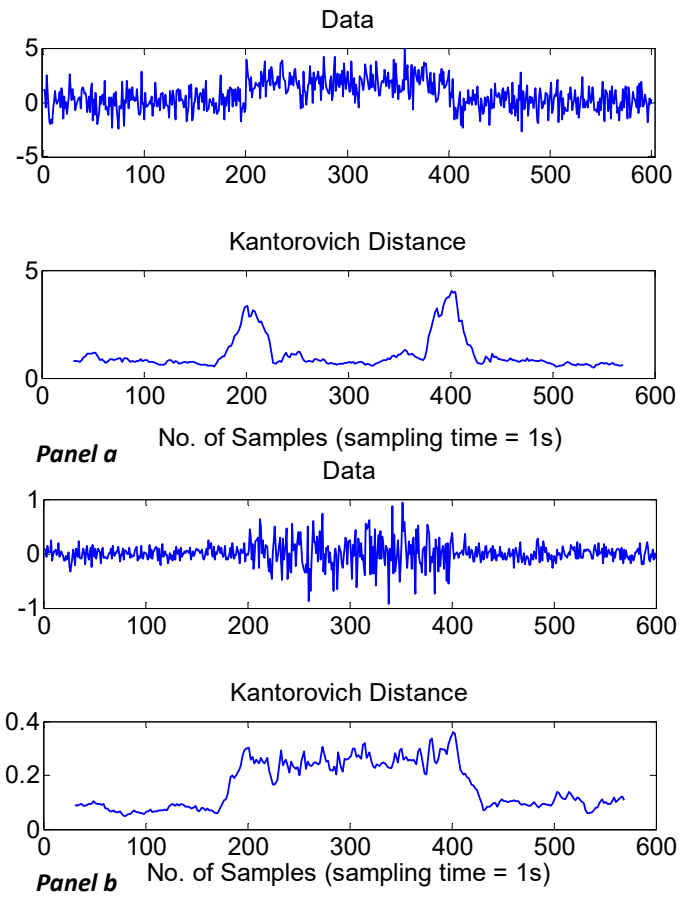


**Figure 2.8:** *a)* Change in mean (0 to 2) with same variance (0.1) along with the calculated Kantorovich distance, *b)* change in variance along with corresponding KD score.

**Change in Variance** Change in variance (second moment) of the data is significant in many cases. The sudden change in the variance of a specific data can be due to increase or decrease in product variability. The latter is preferred since 'less-variability' means more closure to the desired specification. It can also indicate a decrease in measurement noise, increase in perturbation due to a disturbance and so on. So, detecting this type of change may also be helpful in detection of certain types of process improvements or deterioration.

Panel *b* of Figure 2.8 shows the performance of the algorithm on variance change. From the top panel of this figure, a clear increase is observed in the variance of the data in between $200^{th}$ and $400^{th}$ samples. The bottom part of this panel shows the calculated Kantorovich distance for this data indicates higher values between $200^{th}$ and $400^{th}$ samples. The change in variance does not have a large effect as it has in case of the mean shift. But by setting a proper threshold, one can detect the change in variance also which may give important insight into the quality of process operation.

**Detection of Incipient change and Sharp Change** According to the time taken to see the effect of changes, the changes of process industries can be classified into two classes (Gao et al., 2016) are: 1) the step type change or the sharp change and 2) the ramp type or incipient change. In this section, the proposed change detection algorithm was tested with both types of changes.

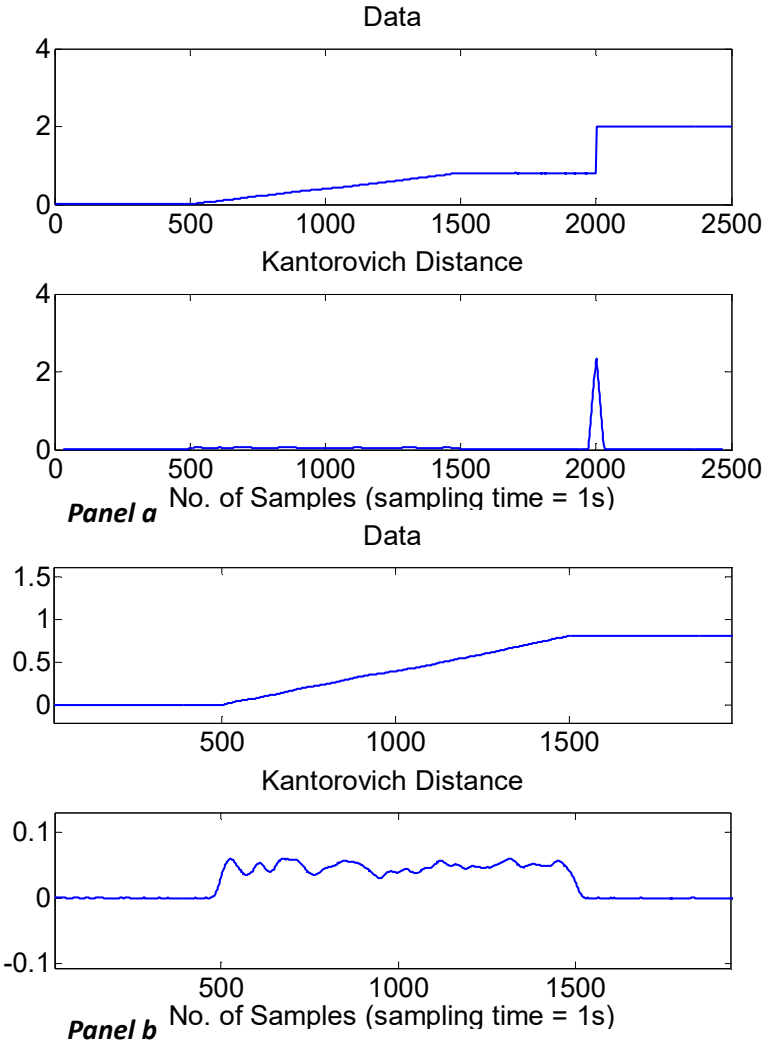


**Figure 2.9:** *a)* Incipient and Sharp change along with the calculated Kantorovich distance, *b)* Zoomed-in version of Panel *a* to show efficacy of KD score in detecting incipient change.

In process industries, one very common type of change is the incipient change which is very small in size and takes relatively longer time to be visible as change. These types of changes remain hidden for a long time until significant damage has occurred. On the other hand, sharp or instantaneous changes are large in size and the effect is visible almost at the same time of the change. Both these types of changes are crucial and may cause

severe damages in product scheduling, product specification, unplanned plant shutdown and sometimes even damage to the plant equipment or injuries of plant personnel.

The proposed change detection scheme can detect both these types of changes. Panel *a* of Figure 2.9 shows both types of changes in the top part. The incipient change occurs from $500^{th}$ to $1500^{th}$ samples and it is obvious that the change is still less than 1 (see y-axis value) after $1500^{th}$ sample. On the other hand, the sharp change occurred at $2000^{th}$ sample and it reached to higher value instantaneously.

The bottom part of this panel shows the corresponding Kantorovich distance for this simulated data. It is seen that both for incipient and sharp change there is a rise in KD score. Though for the incipient change the rise is very small, if this figure is zoomed-in as in panel *b* of figure 2.9, it is obvious the incipient change has also been detected by this algorithm. To detect incipient changes, the threshold value of Kantorovich distance should be lower than that of the sharp change. The choice of the threshold of KD is dependent on the objective. Experience in the process is also valuable while choosing the threshold. To avoid the false alarms due to the noise, a rule of thumb is that the minimum threshold of KD should be at least thrice of the standard deviation of the noise.

### 2.3.2 Change in Simulated Pipeline

The method has been applied on a simulated pipeline with change in operating conditions. The pipeline model as described in section 3.2 is used to generate the simulated pipeline data.

#### Pipeline Description and Simulation Results

In this study, the simulated pipeline is of 40 km in length with 0.4 m inside diameter. For the simulation purpose, the pipeline was divided into 20 equal sections (21 nodes). So, each sec ɔde 1 and valv ɡiven in sectio

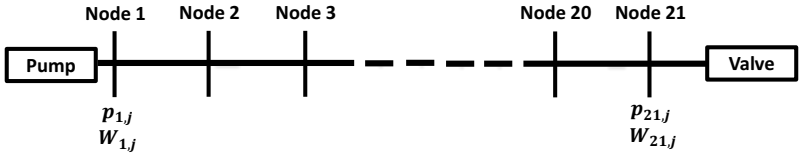


**Figure 2.10:** Schematic diagram of nodes to simulate the pipeline.

To check the efficacy of the change detection method a 10% step-up was introduced

in the first boundary condition i.e. pump at node 1 at $300^{th}$ sample. Then a 5% step-down was introduced in the same boundary condition at $600^{th}$ sample. The simulations were performed for 900 samples. In these simulations, sample time was 1s to meet the Courant-Friederich-Lewy(CFL) condition (Uilhoorn, 2014) because the sonic velocity was calculated as 1185 $\frac{m}{s}$. The working fluid was naphtha.

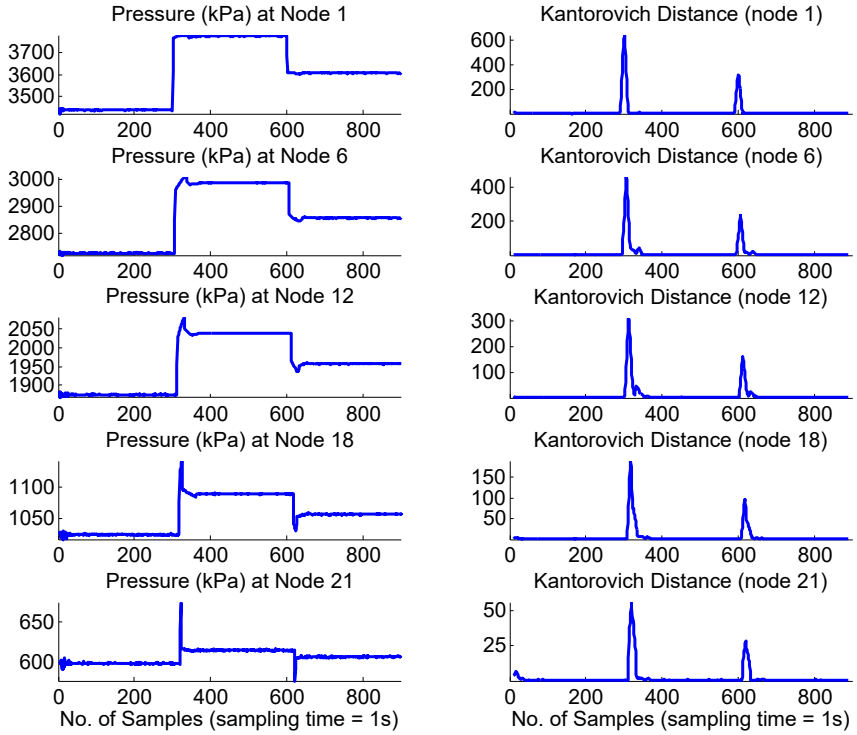


**Figure 2.11:** Change detection results with the proposed algorithm on a simulated pipeline.

These step changes in the first boundary condition will affect the pressure and mass flow-rates of the other nodes also. The Kantorovich distances were calculated for pressures at node 1, node 6 (10 km from the first boundary condition), node 12 (22 km from the first boundary condition), node 18 (34 km from the first boundary condition) and at node 21.

Figure 2.11 shows the simulated pressures (left panels of the figure) of nodes 1, 6, 12, 18 and 21 along with their corresponding calculated Kantorovich distance (right panels of the figure). In these cases, the parameter $m$ was 10 and parameter $k$ was fixed at 2. It is seen in this figure, that the two setpoint changes were detected as soon as the effect of these changes were pronounced in the corresponding nodes. The changes in boundary condition take some time to travel to the downstream nodes and the amplitude of the change in boundary condition is also attenuated while reaching to the downstream nodes.

In a pipeline, pressure reduces along the length of the pipeline due to friction losses

| Nodes | Step-up detection time | Step-down detection time |
|---|---|---|
| 1 | Sample 301 | Sample 601 |
| 6 | Sample 306 | Sample 606 |
| 12 | Sample 312 | Sample 612 |
| 18 | Sample 318 | Sample 618 |
| 21 | Sample 320 | Sample 620 |

**Table 2.1:** Results for offline change detection of the simulated pipeline with step-up and step-down

in the direction of flow. From this figure, it is clear that pressure at node 6 is higher than pressure at node 12 and again pressure at node 12 is higher than pressure at node 18. Hence, pressure at node 1 is the highest and pressure at node 21 is the lowest. The magnitude of the peaks of KD also varied along with the magnitude of the pressure measurements of the corresponding nodes.

From the right panels of Figure 2.11, it is clear that the KD score calculation for the corresponding pressure measurements has started from $11^{th}$ sample which is equivalent to $m + 1$ (minimum sample required to start the offline KD calculation). Table 2.1 shows the sample times corresponding to the peaks of KD for these two changes in the pressure measurements of the relevant nodes. It is noteworthy that with the increase of the distance from the origin of the change (boundary condition at node 1), the detection time (time to reach the corresponding peak of KD) increases. This is because of the travel time required for the pressure wave produced by the change. So, it can be concluded from this section that the proposed method is able to detect changes in case of pipeline operating conditions.

### 2.3.3 Example with Real Pipeline Data

It is always difficult to apply a method to a real pipeline because there are unforeseen and uncontrollable sources of disturbances or process dynamics which make the behavior of the real data different than what is expected. However, the proposed method of change detection has been applied on several real pipelines with changes in operating conditions. All the data analyzed in this section was collected from Suncor pipeline. In the first case, the algorithm was employed to detect sharp changes in the operating condition. In the second case, detection of gradual change on operating condition was studied. For both the cases, representative real offline measurements were fed to the proposed change detection algorithm.

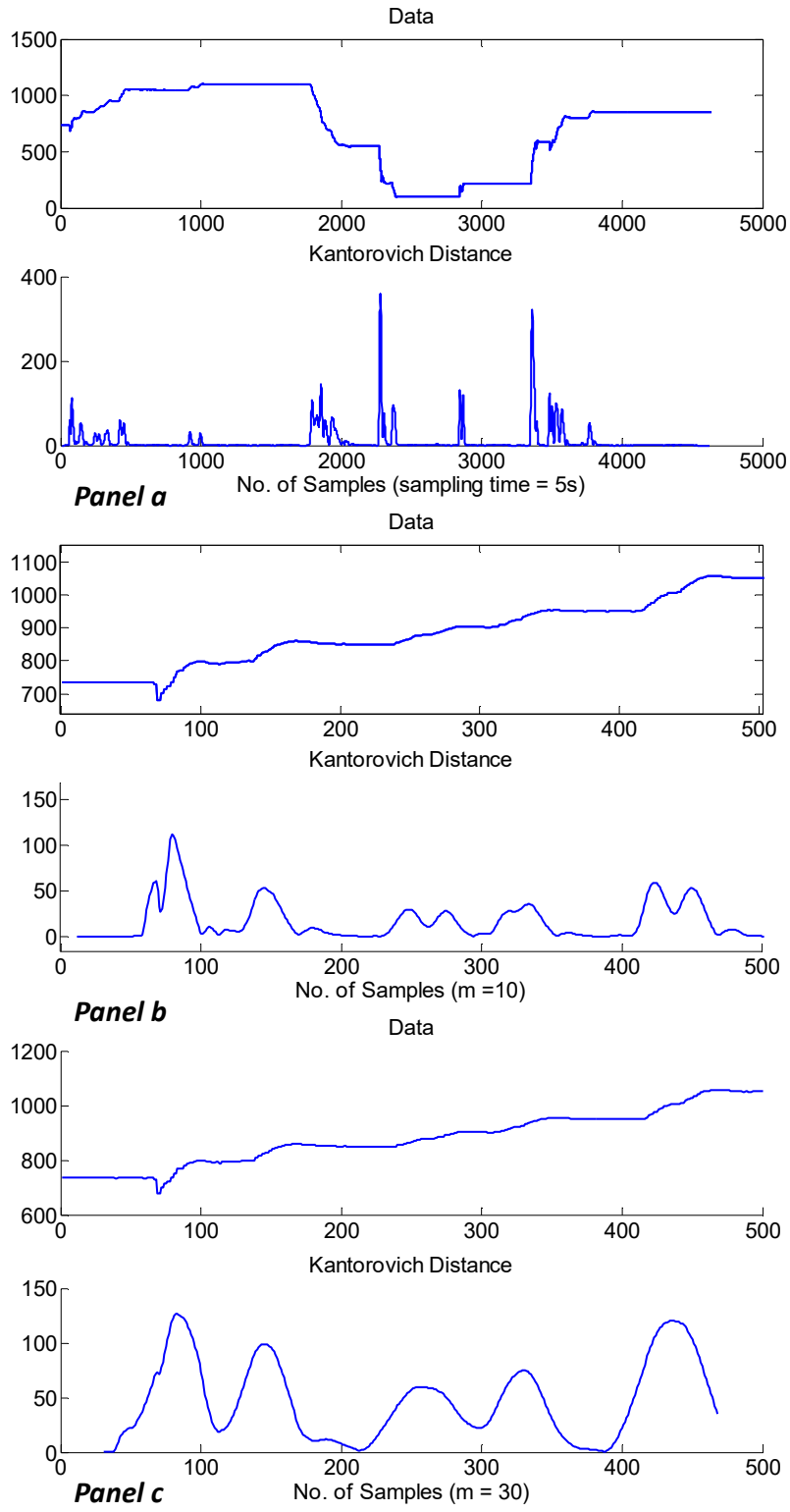


**Figure 2.12:** *a)* Gradual change in operating condition change in real pipeline along with the calculated Kantorovich distance with sampling-time = 5s, *b)* Zoomed-in version of panel *a* for gradual change of first 500 samples with $m = 10$, *c)* Zoomed-in version of panel *a* for gradual change of first 500 samples with $m = 30$

**Detecting Sharp and Gradual Changes in Real Data**

In many processes, gradual change also happens besides the sharp changes. From the earlier discussions, it is clear that the proposed method is efficient enough to detect the sharp changes. In this section, the performance of the proposed method on gradual changes is discussed.

Panel *a* of Figure 2.12 shows the time-series pressure data at a fixed node of a real pipeline. In this case, the sampling-time was 5s. The upper half of the figure shows that there are both sharp and gradual changes in this data. The lower half is the calculated Kantorovich distance for this time series data. It is clear from this figure that the Kantorovich distance associated with sharp changes are higher than that of gradual changes. In this case, two sharp changes occurred around at samples 2273 and 3353. The first sharp change was in downward direction and the second one was in upward direction. The corresponding peaks in Kantorovich distance occurred at 2279 and 3361 samples respectively. Though the peaks started rising from 2243 and 3325 samples respectively.

When there are gradual changes, the proposed change detection scheme considers that a group of small sharp changes. So, a cluster of small peaks are seen in the Kantorovich distance profiles associated with the gradual changes. So, there are two distinctions between the sharp and gradual changes -

1. Unlike the sharp changes, duration of the gradual changes is longer. Sharp change occurs almost instantaneously.

2. Kantorovich distance profile shows a cluster of small peaks in case of gradual changes, whereas in case of sharp changes it shows one sharp peak for the corresponding sharp change.

The evidence of this phenomenon is visible from panel *b* of Figure 2.12. This panel only highlights the first gradual change in between sample 1 to sample 500 and KD was calculated with parameter $m = 10$. From this figure, it is clear how the proposed method works with the gradual changes. From the upper panel of this figure, it is obvious that this gradual change was consisted of 9 small changes. Accordingly, 9 small peaks were found in the Kantorovich distance profile where the peak values varied from 4.6 to 112 which is much lower than the peak value associated with the sharp changes.

Bottom part of this panel of Figure 2.12 shows the calculated KD score for this data. It is obvious that the KD calculation started after at $(m + 1)^{th}$ sample where $m$ is 10 in this case. Parameter $k$ has been fixed as 2 in this case. Table 2.2 summarizes the detected gradual change-points for this real data.

| Peak number | Actual change point | Detection with m=10 | Detection with m=30 |
|---|---|---|---|
| Peak 1 | Sample 66 | Sample 68 | Sample 68 |
| Peak 2 | Sample 71 | Sample 79 | Sample 83 |
| Peak 3 | Sample 138 | Sample 145 | Sample 145 |
| Peak 4 | Sample 240 | Sample 247 | Sample 256 |
| Peak 5 | Sample 266 | Sample 274 | - |
| Peak 6 | Sample 311 | Sample 320 | Sample 330 |
| Peak 7 | Sample 329 | Sample 333 | - |
| Peak 8 | Sample 416 | Sample 423 | Sample 435 |
| Peak 9 | Sample 444 | Sample 450 | - |

**Table 2.2:** Change detection results for a real data with gradual change in offline mode.

In table 2.2, the peaks which are greater than 20 are selected as change points. It is obvious that the first peak corresponds to the step-down at $66^{th}$ sample and the second peak corresponds to the next gradual step-up at $71^{th}$ sample. Since the first step-down was a sharp one it was detected quite early (just 2 sample delay). On the other hand, the second change was a gradual one. So, in this case the change was detected quite later (8 sample delay). There are only slight changes between peaks 4 & 5 and 6 & 7. That is why there are two small adjacent peaks in these cases. Peaks smaller than the value of 10 are due to some very small changes. So, it is very important to choose the minimum threshold to filter out some unnecessary change points. The minimum threshold is very specific to the process, magnitude of the data and the experience of the operator.

The last column of Table 2.2 shows that with $m = 30$, some of the very close peaks are neglected by the algorithm. But on the other hand, in most of cases detection delay has been increased. This is also evident from panel *c* of Figure 2.12. So, there is again a trade-off between early detection and false alarm.

## 2.4 Case Studies with Online/Sequential Change Detection

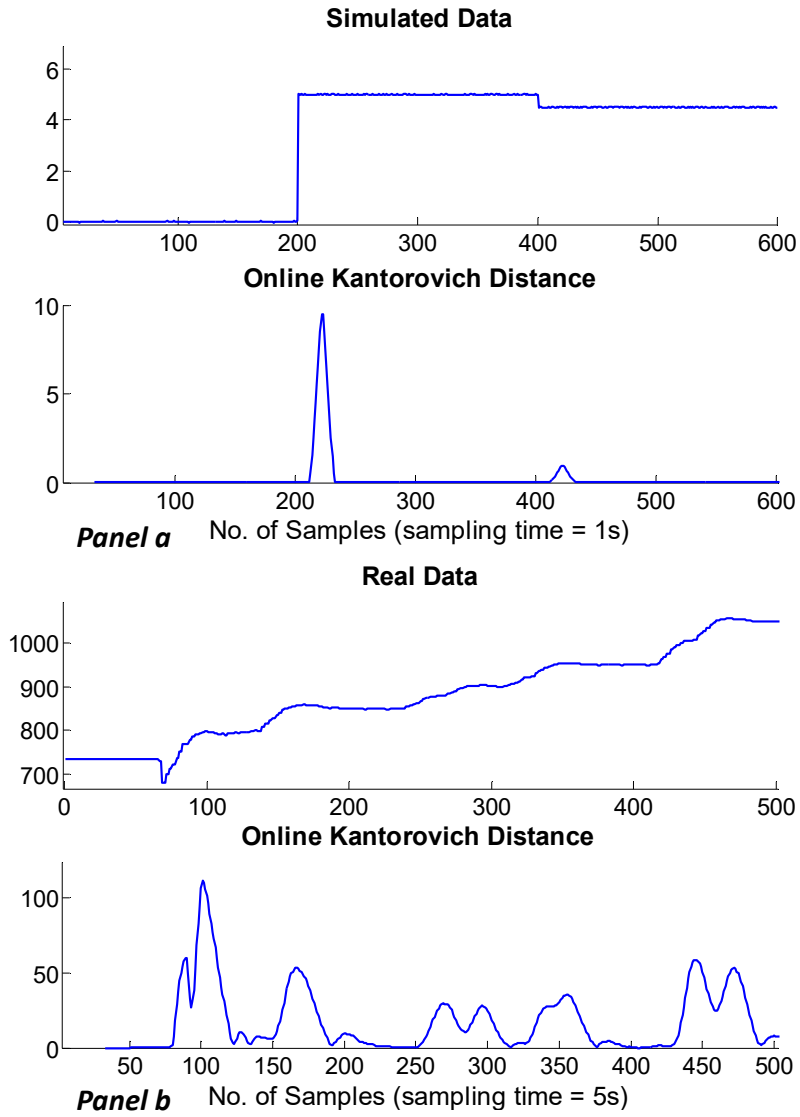


**Figure 2.13:** *a)* Online change detection with simulated data (sampling-time = 1s), *b)* Online change detection with real data (sampling-time = 5s).

In case of online change detection, at least $2m + k - 1$ samples are needed as prerequisite to start the calculation of KD as has been discussed earlier. After the initiation, with the inclusion of a new data point at each sampling instant, the window $2m+k-1$ moves forward one sample. So, in this way the online change detection algorithm calculates KD at each sampling instant after the accumulation of first $2m+k-1$ number of samples. In this section

| **Threshold =** $2\sigma$ (0.02) | **Threshold =** $3\sigma$ (0.03) | **Threshold =** 0.5 | **Threshold =** 1 |
|---|---|---|---|
| Sample 71 | Sample 222 | Sample 222 | Sample 222 |
| Sample 152 | Sample 422 | Sample 422 | - |
| Sample 198 | - | - | - |
| Sample 200 | - | - | - |
| Sample 222 | - | - | - |
| Sample 243 | - | - | - |
| Sample 253 | - | - | - |
| Sample 256 | - | - | - |
| Sample 316 | - | - | - |
| Sample 320 | - | - | - |
| Sample 422 | - | - | - |

**Table 2.3:** Choice of threshold for online change detection

two case studies with online change detection will be discussed. The first one is simulated data, the second one is an application to real data.

### 2.4.1 Online Change Detection with Simulated Data

For this case study, simulated data with one large and one small step changes were generated. Total sample length was 600 with sampling-time 1s. The first step change was introduced at $200^{th}$ samples by increasing mean of the data from 0 to 5. The second step change was 10 time smaller and it was a step-down. The step-down of magnitude 0.5 was introduced at $400^{th}$ samples. This time the mean was changed from 5 to 4.5. The standard deviation of the noise for this data set was 0.01 ($\sigma$).

Top part of panel *a* of Figure 2.13 shows the generated data and bottom part of this panel shows the corresponding Kantorovich score. MATLAB built-in function *findpeaks* was used to find the peak of Kantorovich distance for online case. The most crucial parameter for online detection is the defining the minimum threshold. If the threshold is too low, due to effect of noise many false alarms can be raised. On the other hand, if the threshold is increased too much, real changes could be missed (missed alarm). Table 2.3 gives an idea about the choice of threshold. Good knowledge about the process specifically the concerning data-tag on which KD is being calculated, can help to get an accurate choice of minimum threshold. It also depends upon how much sensitivity is wanted from the algorithm. From Table 2.3 it is seen that if the least threshold was chosen below the $3\sigma$ then there were 9 false alarms whereas there were only two real changes. On the other hand, if the least threshold was chosen arbitrarily large as 1, even the real step-down at $400^{th}$ sample was missed by the algorithm which was smaller in magnitude comparing to

| Peak number | Actual change point | Offline change point | Online change point |
|---|---|---|---|
| Peak 1 | Sample 66 | Sample 68 | Sample 90 |
| Peak 2 | Sample 71 | Sample 79 | Sample 101 |
| Peak 3 | Sample 138 | Sample 145 | Sample 167 |
| Peak 4 | Sample 240 | Sample 247 | Sample 269 |
| Peak 5 | Sample 266 | Sample 274 | Sample 296 |
| Peak 6 | Sample 311 | Sample 320 | Sample 342 |
| Peak 7 | Sample 329 | Sample 333 | Sample 255 |
| Peak 8 | Sample 416 | Sample 423 | Sample 445 |
| Peak 9 | Sample 444 | Sample 450 | Sample 472 |

**Table 2.4:** Change detection results for real data in online mode.

the change at $200^{th}$ . So, there is always a trade of between the false alarm rate (FAR) and missed alarm rate (MAR). Often this is explained by receiver operating curve (ROC) in the literature of detection (Levy, 2008).

For this example, it can be concluded to choose the minimum threshold as greater than $3\sigma$ but not more than 0.5. It varies from process to process and with the objective of the change detection. If it is wanted to detect really small changes, the threshold should be very small. On the other-hand, if it is aimed to detect large changes only, the threshold should be high.

However, as expected the changes were detected little bit late as this is an example of online change detection. For large step-up, the change was detected at 222 sample and for the small step-down the change was detected at 422 samples.

### 2.4.2 Online Change Detection with Real Data

In this case, the same real data is used as in section 2.3.3 but the detection mode was online. Panel *b* of Figure 2.13 shows first five hundred samples of this real data-set as in panel *b* of Figure 2.12. The top part of this panel shows the real data and bottom part of this pane is the corresponding calculated KD scores in online mode.

Table 2.4 summarizes the result for this online detection. In this case also, the peaks smaller than 20 in the KD score were neglected. Column 2 of this table shows the actual initiation time of the corresponding changes. Column 3 represents the change points detected by the proposed algorithm in offline mode. Column 4 gives the change points detected in online mode. In case of both column 3 and 4, the value for parameter $m$ was 10 and parameter $k$ was 2. From table 2.4, it is clear that for online detection the change points are detected later than the offline detection. This is because, for online detection the algorithm must wait till the accumulation of $2m + k - 1$ samples before the first instance of

KD is calculated.

## 2.5 Online Change Detection with Benchmark Tennessee Eastman (TE) Process Data

In this study, the modified Tennessee Eastman process as described by Bathelt et al. (2015) is used to check the efficacy of the proposed change detection algorithm in real-time mode.

According to the time taken to see the effect of changes, the changes of chemical process industries can be classified into two classes (Gao et al., 2016) are: 1) the step type change or the sharp change and 2) the ramp type or incipient change. In this section, both the sharp and incipient types of changes were tested. The simulation time horizon is 1300 hours. The changes were introduced by modifying 9 of the total twelve 12 manipulated variables of the TE process (Bathelt et al., 2015).

Ramp type changes were introduced for the $6^{th}$ manipulated variable at 600 hr and it was there until the end of the simulation. Step changes were introduced on manipulated variables number 1, 2, 3, 4, 5, 7, 8 and 9 at 100, 200, 300, 400, 500, 900, 1000 and 1100 hrs respectively.

Figures 2.14 and 2.15 show the effect of these changes on some of the representative measurements available from the TE process (Bathelt et al., 2015). There are total 41 measurements for the TE process. Figures A.1 to A.4 show the effect of these changes on the rest of measurements available from the TE process (Bathelt et al., 2015). Right-hand sides of these figures show the corresponding KD calculation. The usual sampling-time of TE process is 0.01 hr. The data was first down-sampled to 10 times of the original data to reduce the computational burden. At the next stage, an exponentially weighted moving average (EWMA) filter was used to filter out the noise from the data. The fraction of true measurement used while filtering was 0.005. KD based change detection algorithm was applied to this filtered data. The parameter $m$ was 100 in all these cases and parameter $k$ was fixed as 5.

It is seen from the above-mentioned figures that the KD algorithm was successful in detecting all the sharp changes those were introduced at 100, 200, 300, 400, 500, 900, 1000 and 1100. The detection has some delay because to detect the online change, $2m + k - 1$ samples to be accumulated to start the KD calculations.

Fifth ($5^{th}$) panel of Figure 2.15 shows that the ramp change is also detected by the proposed algorithm for the measurement 'Component G in Product, Mole%'. The ramp change effect is also visible in KD calculation of other measurements, but comparing to the KD value related to the sharp changes, the effect of ramp change is not sufficiently

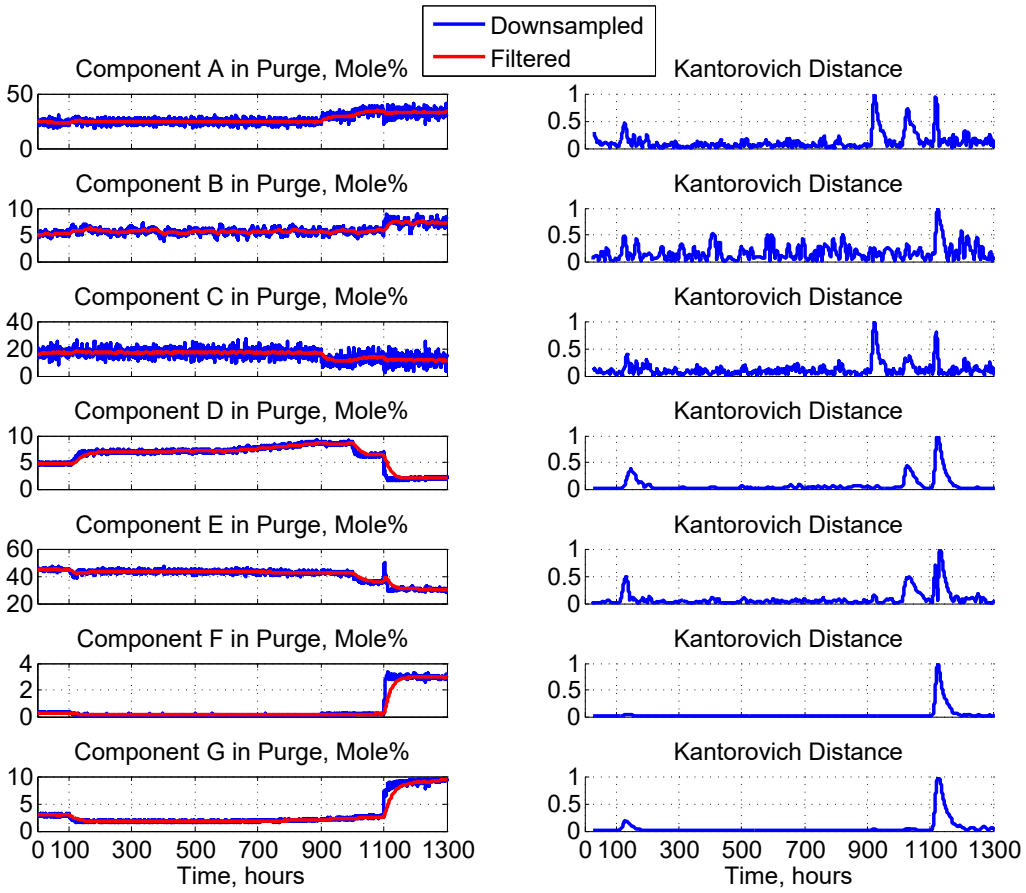


**Figure 2.14:** Measurements 29 to 35 from TE process and their corresponding normalized KD scores.

pronounced. If the appropriate portions of the KD score profile is zoomed in, this claim can be justified. For example, if the $4^{th}$ panel of Figure 2.14 is zoomed in, corresponding to the measurement 'Component D in Purge, Mole%', it is visible that the ramp change has been detected with higher KD score (Figure 2.16) after $600^{th}$ sample. From the bottom panel of Figure 2.16, it is clear that mean of KD score is higher for $600 - 900$ hr than that of $300 - 600$ hr.

Figure 2.17 shows the overall change points detected from all the measurements generated from the TE process. This plot combines the normalized KD score from all the 41 available measurements and give the overall normalized KD score. Hence, this figure provides the comparison of all the changes and the higher score in this figure indicates that the corresponding changes have a higher impact on the process. That is why the changes which affected only a few of the 41 measurements such the ramp change at $600^{th}$ or the step changes at $300^{th}$ hr of the original data has smaller peaks. On the other hand, step changes

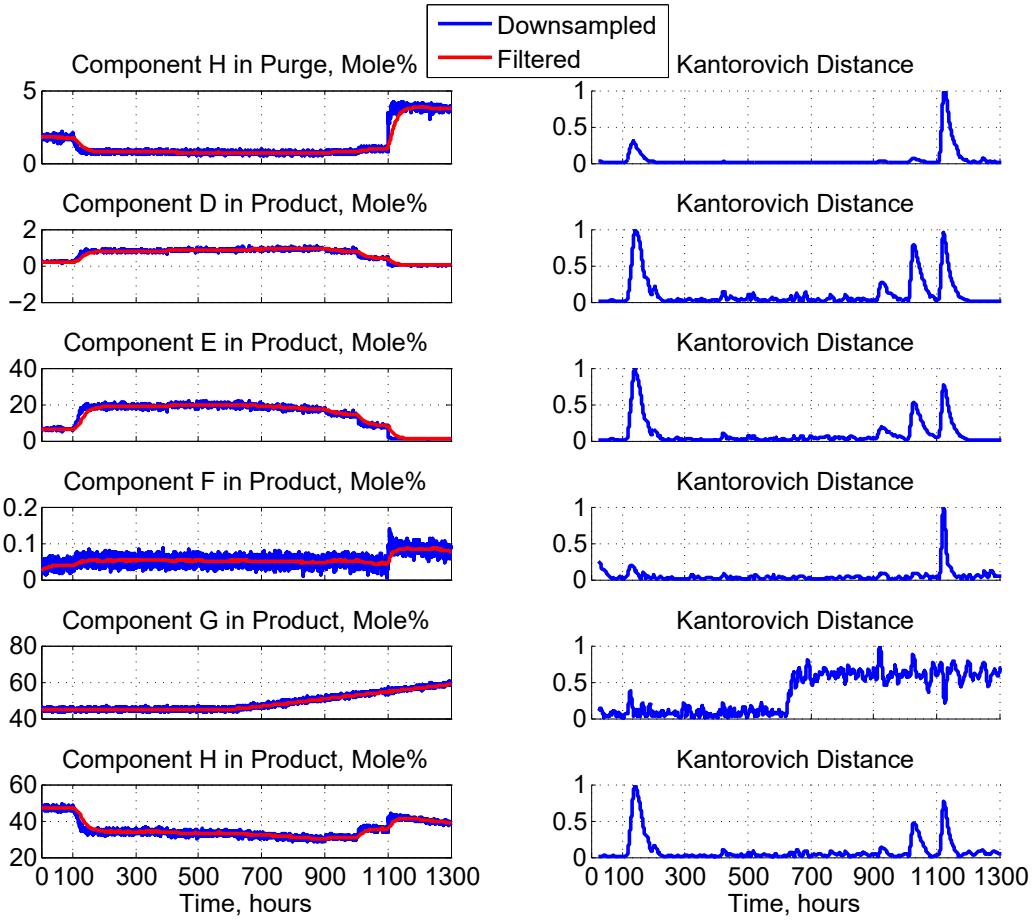


**Figure 2.15:** Measurements 36 to 41 from TE process and their corresponding normalized KD scores.

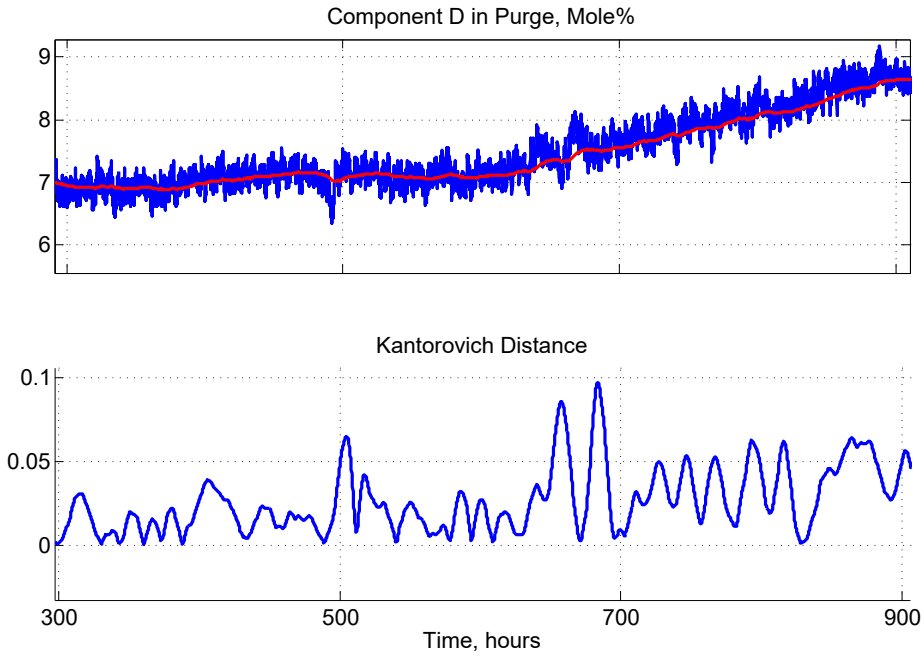


**Figure 2.16:** Zoomed-in version of the 'Component D in Purge, Mole%' and its corresponding KD score.

at $100^{th}$ hr, $400^{th}$ hr, $500^{th}$ hr, $900^{th}$ hr, $1000^{th}$ hr and $1100^{th}$ hr have higher impacts that means these changes were pronounced in most of the available measurements. So, these changes have an impact on the overall process.

It is noteworthy that few hours after $600^{th}$ hr of the imputed data, the base of normalized KD shifted from the black line to the red line in Figure 2.17. This is because, at $600^{th}$, the ramp (incipient) change was initiated. This change is very slow in nature, but it also has an effect on the normalized KD. Hence, the base of KD score kept increasing due to this change. The ramp-type change was continued till $1300^{th}$ hr of the imputed data. Since the ramp-type change was still going on, one can see that the base of KD score is increasing above the red line from $1100^{th}$ hr and onward. The step changes at $900^{th}$ hr, $1000^{th}$ hr and $1100^{th}$ hr had added effect with this ramp change.

This change-detection information can be used in various ways. If the user wants to focus only on the smaller changes, the threshold should be low. On the other hand, if the changes that create a higher impact on the overall process are to be identified, the threshold should be higher to filter out the smaller changes.

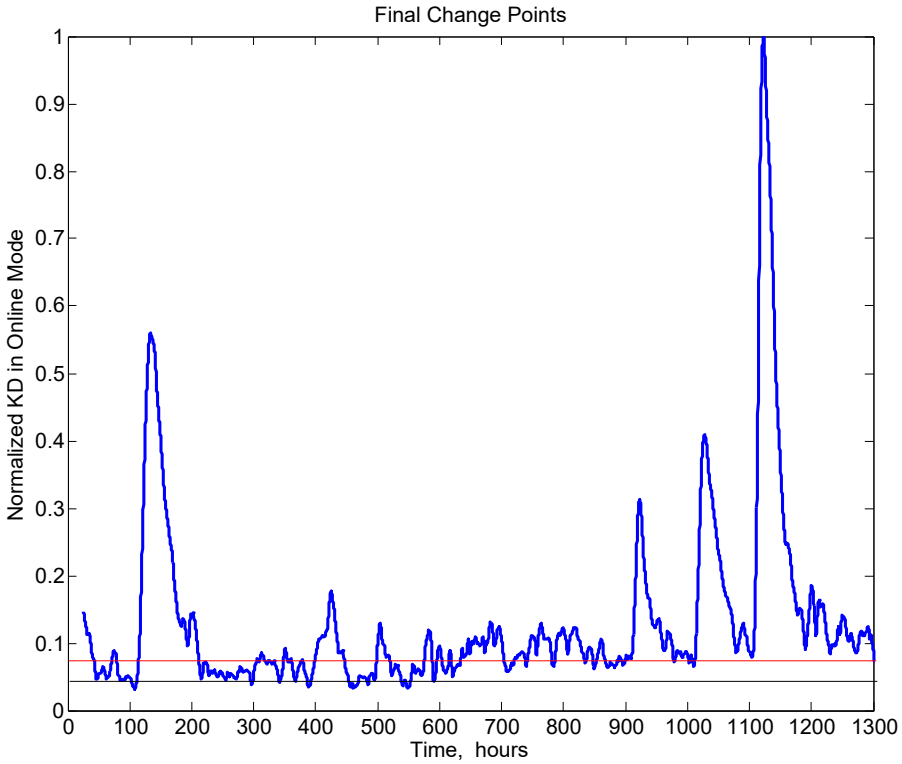


**Figure 2.17:** Overall changes from TE process.

## 2.6 Concluding Remarks

In this chapter, a novel change detection algorithm based on Kantorovich distance method has been proposed. The algorithm was applied to both simulated cases and real data in both online and offline modes. The proposed algorithm was also applied to the benchmark Tennessee Eastman process in real-time mode. This change detection algorithm will help detect the process operating condition change, sharp and gradual (incipient) change during the process start-up and shutdown and will aware the operator about the change. Incorporating this information into the existing process monitoring algorithms, alarm-floods events can be handled efficiently during a transient in the process. The idea could be to turn off all the alarms when a normal or desired change or transition in the process is detected by this algorithm, but this needs further analysis.

Results with simulated and real pipeline show that this method can detect changes in process plants. This method is efficient enough to detect both sharp and gradual changes. Simulation and real data results with online mode also show the efficacy of the proposed method when it is being applied online.

# Chapter 3

# Modeling Pipeline Leak and Leak Signature

## 3.1 Introduction on Pipeline Leak

Pipelines are now commonly used to transport hydrocarbon fluids over long distances, from the production site to the end-user. These pipelines can suffer from leaks due to age, corrosion or damage by a third party (Zhang et al., 2015). Moreover, the fluids are often flammable, toxic, corrosive and hazardous to the environment. Therefore, a leak in such a pipeline can cause loss of valuable products and energy, serious pollution to the environment, an injurious impact to human health and property as well as operational downtime. The owner of the pipeline not only faces the prospect of a bad reputation in media but also financial losses regarding environmental cleanup fees, fighting lawsuits plus the operational losses. Hence, early detection and localization of leak is of utmost importance.

Existing pipeline leak detection and localization systems (LDS) can be broadly classified as external monitoring or hardware based methods and internal monitoring based or software based methods (Geiger, 2006; Murvay and Silea, 2012). Figure 3.1 shows different classes of leak detection systems illustratively. There is also a third class called the non-technical methods which refers to the visual inspection, smelling the tracer element, sensing special sound and soap bubble screening method (Murvay and Silea, 2012). These methods include trained personnel or a specially trained dog along with its trainer, to walk along the pipeline or inspection of the pipeline from a manned or unmanned aircraft for visual or other evidence of leaks. The downside of these methods is that detection depends upon the frequency of the inspection. The accuracy of the method is also dependent on the experience and skill of the inspector. Finally, buried pipelines are not accessible for visual inspection.

External or hardware based methods work on the principle of physical detection of

escaping fluid using local sensors. External methods can usually detect the leak location more precisely, but they are more costly and cannot easily and economically be retrofitted in existing buried pipelines, in most cases (Geiger, 2006). Generally, the testing cycle of external methods is longer than that of software-based methods (Zhang et al., 2015). Due to the complexity to apply and high cost, the external methods are not generally used for continuous monitoring of leaks on long pipelines.

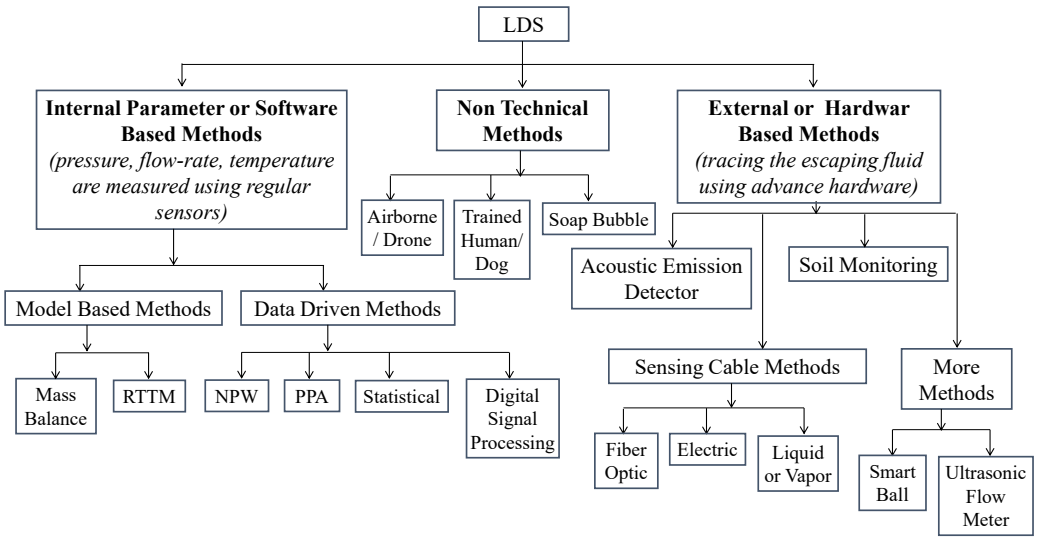


**Figure 3.1:** Classification of leak detection systems.

One of the popular classes of external leak detection system is the acoustic emission detector type method (Watanabe et al., 1993) and its variants. The basic idea is that the leaked fluid leaves behind an acoustic signal while product passes through a leaking hole in the pipe. Acoustic sensors, placed at the two ends of the pipeline, detect this acoustic signal and the leak is localized based on the time difference required to reach the signal at the two ends of the pipeline. However, though these methods are gaining much attention in recent years for leak localization, they often suffer from inaccuracy due to the presence of other sources of extraneous noise such as a pump. The accuracy is also dependent on the range of detection and frequency of the acoustic sensors.

The fiber optic sensing cable method is one of the most sensitive hardware-based methods. In this method, a fiber optic cable is installed along the entire length of the buried pipeline. The cable is initially at the same temperature as the ground. When the leaked fluid comes into contact with the cable, the temperature of the cable changes in the leakage position (Geiger, 2006). Depending upon the active or passive mode, this change in temperature is detected either by sensing significant absorption or scattering of the emitted

radiation by the leaked fluid molecule (active) or utilizing thermal imaging technique (passive) (Murvay and Silea, 2012). Thus, leak is detected and localized. However, this method is very costly and works only for pipelines which carry products which are either warmer or colder than ground temperature.

Other sensing cable methods such as the electrical, liquid or vapor sensing cables are also based on burying an auxiliary sensing cable along the full length of the main pipeline. In case of the electrical cable, the materials are chosen such that some of its electrical properties (resistance, capacitance etc.) will change when it comes into contact with the fluid passing through the main pipeline. Similarly, the liquid or vapor sensing cables are permeable to the fluid passing through the main pipe (Geiger, 2006). Hence, in the occasion of a leak, the target fluid comes into contact with the sensor and some properties of the sensors are changed. Utilizing appropriate methods to detect these changes in the sensors, a leak is detected and localized. The accuracy of all these sensor-based methods are high, but in general, the cost of implementing them is also very high and it is very difficult to employ them in existing pipelines. Moreover, the electric cable sensor cannot estimate the size of the leak and the liquid and vapor sensing cables cannot be installed for above the ground or high depth pipeline systems (Murvay and Silea, 2012).

In case of the soil monitoring method, a non-hazardous tracer element is injected into the pipeline along with the desired product. By detecting this tracer chemical in the soil, a leak is detected and localized accurately. However, the downside of this method is that the pipeline should be underground. Continuous injection of the tracer chemical and its detection mechanism make the method very costly (Murvay and Silea, 2012).

There are also some recently developed hardware-based methods such as the ultrasonic flow meter (Murvay and Silea, 2012) and the smart ball (Fletcher and Chandrasekaran, 2008) methods. The entire pipeline is divided into series of segments in the ultrasonic flow meter based method. At both ends of each of these segments there will be site-stations which will calculate volumetric flow-rate of the fluid, sonic velocity, fluid and ambient temperature using the flow and temperature measurements. The master station combines all the data collected by those site-stations and computes volume balance by comparing the input and output end volumetric flow-rates of each of those segments. This method can localize a leak with an accuracy of 150 m. Once again, this method may be cost prohibitive as compared to other leak detection methods, due to the significant number of sensors required. It is also difficult to be retrofitted to buried pipelines (Murvay and Silea, 2012).

The smart ball is a moving acoustic sensor which travels through the pipeline along with the fluid and continuously collects the acoustic signals. So, it offers good accuracy of detection and localization even with a small leak. The main disadvantage with this

technology is that background noise of nearby pumps or other devices may result in false detection. Moreover, the detection is not done until the ball exits at the pipeline end and all the stored acoustic signals are analyzed (Fletcher and Chandrasekaran, 2008). Therefore, this method takes some time to analyze the data and confirms there either was or wasn't a leak during the specific time the ball was in the pipeline. Continuous smart ball operation would be required for ongoing leak detection.

The other large class of LDS systems, the internal or software based methods can be broadly divided into two groups: 1) model-based methods and 2) data-driven methods. Internal methods monitor pipeline parameters using regular pipeline sensors such as the pressure, mass flow-rate or temperature sensors. Hence, the software-based methods are quite easy to apply for continuous monitoring with less cost. But most of them cannot detect the leak location precisely, especially for small leaks. Additionally, some of these methods take a longer time to detect small leaks. If these leak detection methods are sensitized to detect small leaks, there is always the issue that the operators are overwhelmed by too many false alarms (Al-Rafai and Barnes, 1999).

The simplest software-based methods are the volume or mass balance methods (Fukushima et al., 2000). These methods are based on the law of mass conservation. In the case of incompressible fluids, density does not change with the change of pressure and temperature. So, the difference between inlet and outlet flow-rates will be equal to the rate of change of packing or inventory volume. When the input flow is higher than the output flow, it can be assumed that a leak has occurred in the pipeline (Di Blasi and Muravchik, 2009). These methods are simple and low cost. However these methods cannot provide an estimation of leak location. Therefore, another LDS method needs to be employed in parallel for leak localization. Uncertainties in the calculation of packing volume may decrease the reliability of these types of method.

The real-time transient model (RTTM) based method incorporates pipeline model consisting of the continuity and momentum equations to calculate the response of the pipe system in normal conditions (Billmann and Isermann, 1987; Arifin et al., 2015). The model is represented by a set of coupled nonlinear hyperbolic partial derivative equations (PDE). The analytical solution of this PDE system is not feasible. So, in most of the cases, the system is first discretized by using the method of characteristic (Thomas, 1999) or other numerical techniques. In some studies, state estimators (Reddy et al., 2011; Arifin et al., 2015) are used along with the pipeline model to improve the accuracy of estimates especially when the amount of available measurements is less. These estimated values are then compared with available measurements and if the residuals between the estimated and real measurement are greater than pre-specified threshold values then an abnormality is

detected. However, the performance of these methods is highly dependent on the accuracy of model parameters and sensors. The model parameters suffer from uncertainties due to changes or imprecise information on fluid properties (density and viscosity), fluctuation in ambient and process conditions and changes in pipeline properties (scaling, roughness and grade changes). Moreover, these methods also require high computational loads to solve this complex non-linear model. RTTM systems are functional, but they require significant tuning within the initial start-up period and may require continuous adjustments over the life of the pipeline.

The negative pressure wave (NPW) method is a type of internal data-driven method which does not require any model and extra hardware except for already installed pressure sensors. A negative pressure wave will propagate in both the upstream and downstream directions from the point of the leak with sonic (wave) velocity of the working fluid in the pipeline. The leak is localized by observing the time difference of the the negative pressure wave (NPW) to reach the farthest upstream and downstream nodes (Silva et al., 1996). However, the accuracy of leak location is dependent on the number of transmitters at either end of the line. A disadvantage is that this does not work during pigging operations as the wave would be blocked. It also does not work through intermediate pump stations.

Pressure point analysis (PPA) methods try to detect a leak by continuously monitoring pressure measurements and utilizing the fact that the pressure drops when a leak occurs. A leak alarm is raised when the mean values of the available pressure measurements decrease below a preset threshold (Murvay and Silea, 2012). These methods are simple since they do not need any model. However, these methods are not reliable during transient flow conditions.

Statistical methods try to capture the signature of the leak from the available process measurements such as flow-rate and pressure. Applying suitable statistical analysis on these measurements, a leak can be detected and localized. In Zhang (1997), such a statistical leak detection method has been proposed based on Wald's sequential probability ratio test (SPRT). This method works quite well even with the transient processes both for leak localization and detection. The downside of this method is two-fold. Firstly, tuning the parameters of the algorithm requires at least two weeks of past normal condition (no leak) data. If there is a leak while tuning the parameters, this method cannot detect a leak of that size unless it grows (Murvay and Silea, 2012). Secondly, this method suffers from poor localization performance when the leak size is less than 10%. Willis (2011) introduced a modified SPRT based method which is claimed to have low false alarms, but it cannot localize leak.

A similar approach as the statistical method is the digital signal processing method. The

only difference is that digital signal processing tools are used to capture the leak signature from the noisy data instead of the statistical tools. This method also requires leak free data during the parameter tuning stage. Another downside of this method is that it is difficult to implement (Murvay and Silea, 2012).

## 3.2 Pipeline Simulation Model

The method has been applied on a simulated pipeline with change in operating conditions. To generate the simulated pipeline data, discretized version of coupled hyperbolic continuity (equation 3.1) and momentum (equation 3.2) equations were used. Discretization was done using the method of characteristic (Thomas, 1999). Equations 3.3 and 3.4 are the discretized version for this system. While discretizing, the Courant-Friederich-Lewy(CFL) condition (Uilhoorn, 2015) was kept satisfied by taking the fluid sonic velocity in the pipeline ($C_{son}$) as $\frac{\Delta x}{\Delta t}$. The details derivation of the momentum and continuity equations are given in Appendix B.

$$\frac{\partial p}{\partial t} + \frac{C_{son}^2}{A}\frac{\partial W}{\partial x} = 0 \tag{3.1}$$

$$\frac{1}{A}\frac{\partial W}{\partial t} + \frac{\partial p}{\partial x}(1 - \frac{\nu^2}{C_{son}^2}\frac{W^2}{A^2}) + 2\nu\frac{W}{A^2}\frac{\partial W}{\partial x} + \frac{2\nu W|W|}{A^2 D}f + \frac{g}{\nu}sin\theta = 0 \tag{3.2}$$

The discretized equations are -

$$p_{i,j} = \frac{1}{2}(p_{i-1,j-1} + p_{i+1,j-1}) + \frac{C_{son}}{2A}(W_{i-1,j-1} - W_{i+1,j-1}) + \Delta t\frac{C_{son}}{2}\frac{2f}{A^2 D}(\nu_{i+1,j-1}W_{i+1,j-1}|W|_{i+1,j-1} - \nu_{i-1,j-1}W_{i-1,j-1}|W|_{i-1,j-1}) + \Delta t\frac{C_{son}}{2}gsin\theta(\frac{1}{\nu_{i+1,j-1}} - \frac{1}{\nu_{i-1,j-1}}) \tag{3.3}$$

$$W_{i,j} = \frac{1}{2}(W_{i-1,j-1} + W_{i+1,j-1}) + \frac{A}{2C_{son}}(p_{i-1,j-1} - p_{i+1,j-1}) + \Delta t\frac{A}{2}\frac{2f}{A^2 D}(\nu_{i+1,j-1}W_{i+1,j-1}|W|_{i+1,j-1} + \nu_{i-1,j-1}W_{i-1,j-1}|W|_{i-1,j-1}) + \Delta t\frac{A}{2}gsin\theta(\frac{1}{\nu_{i+1,j-1}} + \frac{1}{\nu_{i-1,j-1}}) \tag{3.4}$$

where

- $i$ = node index
- $j$= time index

- $p_{i,j}$ = pressure at node $i$ and time $j$ $(Pa)$
- $C_{son}$ = velocity of sound in the fluid inside the pipe $(\frac{m}{s})$
- $A$ = cross-sectional area of the pipe $(m^2)$
- $W_{i,j}$ = mass flow-rate at node $i$ and time $j$ $(\frac{kg}{s})$
- $f$ = pipeline friction factor (Fanning friction factor)
- $D$ = pipeline diameter $(m)$
- $\triangle t$ = sample time $(s)$
- $\nu_{i,j}$ = specific volume at node $i$ and time $j$ $(\frac{m^3}{kg})$
- $\rho_{i,j}$ = fluid density at node $i$ and time $j$ $(\frac{kg}{m^3})$
- $g$ = gravitational acceleration $(\frac{m}{s^2})$
- $\theta$ = angle of elevation from the flat land

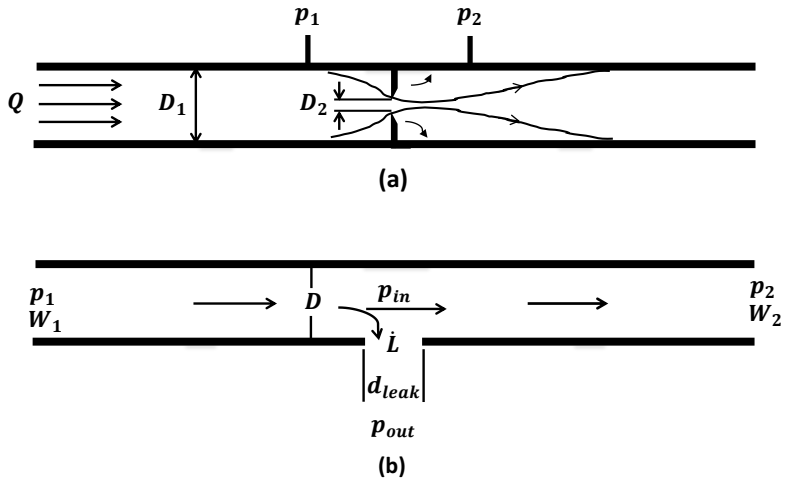


**Figure 3.2:** Panel ($a$) shows the the schematic diagram of an orifice meter and panel ($b$) depicts a typical leak schematically.

In this study, a leak was simulated by modified orifice equation (Ge et al., 2008) along with the pipeline model introduced in section 3.2. Panel ($a$) of Figure 3.2 shows a typical orifice

meter that is used to measure the flow-rate in a pipeline. Assuming that there is no elevation change, the orifice equation is as follows (Daugherty and Franzini, 1977):

$$Q = C_d \frac{\pi}{4} D_2^2 \sqrt{\frac{2(p_1 - p_2)}{\rho(1 - \beta^4)}} \tag{3.5}$$

where

- $D_1$ = Inside diameter of the pipe
- $D_2$ = Orifice diameter
- $C_d$ = Discharge coefficient
- $\rho$ = Fluid density
- $\beta$ = Diameter ratio ($\frac{D_2}{D_1}$)
- $p_1$ = Upstream Pressure
- $p_2$ = Downstream Pressure
- $Q$ = Volumetric flow-rate

Equation 3.5 can be modified to give the mass flow-rate rather than the volumetric flow-rate:

$$\dot{m} = C_d \rho \frac{\pi}{4} D_2^2 \sqrt{\frac{2(p_1 - p_2)}{\rho(1 - \beta^4)}} \tag{3.6}$$

Comparing panels (*a*) and (*b*) of Figure 3.2, one can see that there are two significant differences between the orifice flow and a pipeline leak : a) in case of orifice, the direction of flow inside the orifice and the pipeline is same, whereas in case of leak they are perpendicular, b) unlike the orifice diameter ($D_2$), the leak diameter ($d_{leak}$ in Panel (*b*) of Figure 3.2) is unknown while calculating the leak rate ($\dot{L}$).

Therefore, some modifications were made before applying the orifice equation in case of a leak. It was assumed that in this case, $p_{out} = p_{atm}$.

From Panel (*b*) of Figure 3.2, it can be assumed further that $p_{in} \approx \frac{p_1 + p_2}{2}$. Hence the rate of leak can be defined as following:

$$\dot{L} = C_L \sqrt{p_{in} - p_{atm}} \tag{3.7}$$

where $C_L$ is the leak coefficient. Its value depends upon the leak diameter ($d_{leak}$) which is unknown while detecting a leak. Equation 3.7 can be further simplified by noting that $p_{in} - p_{atm} = p_{in,gauge}$ as follows :

$$\dot{L} = C_L \sqrt{p_{in,gauge}} \tag{3.8}$$

The discretized set of equations with leak at space-node $i$ can be written by combining equations 3.8, 3.9, 3.10 and 3.11. Equation 3.9 refers to the mass flow-rate of the immediate next node after the leak, equation 3.10 refers to the mass flow-rate of the immediate previous node of the leak. Equation 3.11 represents the modified pressure estimation at the leak node due to the leak of size $\dot{L}$.

$$W_{i,j} = W_{i+1,j-1} + \frac{A}{C_{son}}(p_{i,j-1} - p_{i+1,j-1}) - \Delta t \frac{f}{2AD} \nu_{i+1,j-1} W_{i+1,j-1} |W|_{i+1,j-1} - \Delta t A g sin\theta \frac{1}{\nu_{i+1,j-1}} - \frac{\dot{L}}{2} \tag{3.9}$$

$$W_{i-1,j} = W_{i-2,j-1} + \frac{A}{C_{son}}(p_{i-1,j-1} - p_{i-2,j-1}) - \Delta t \frac{f}{2AD} \nu_{i-2,j-1} W_{i-2,j-1} |W|_{i-2,j-1} - \Delta t A g sin\theta \frac{1}{\nu_{i-2,j-1}} + \frac{\dot{L}}{2} \tag{3.10}$$

$$p(i,j) = p(i,j) - \frac{\dot{L} C_{son}}{2A} \tag{3.11}$$

The above pipeline simulation model was implemented in Matlab to generate simulated pipeline data under leak condition (see section 4.3.3). The proposed LDS algorithm was first tested using those simulated data.

Figures 3.3 and 3.4 are showing the simulated leak at node 13 of the pipeline simulated as in section 4.3.3.

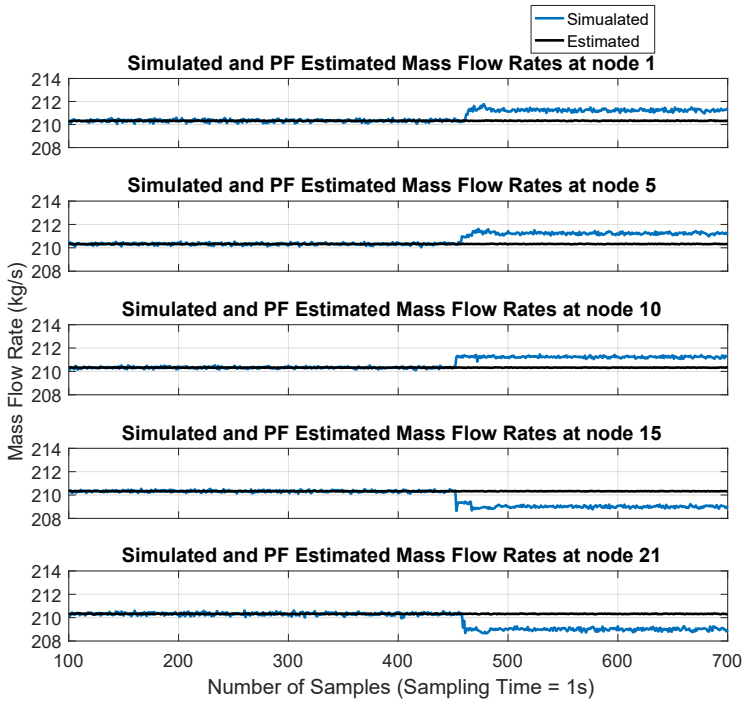


**Figure 3.3:** Residuals between the PF estimated mass flow-rates and available true (simulated) flow-rates for 5% leak at node 13 at 450 s.

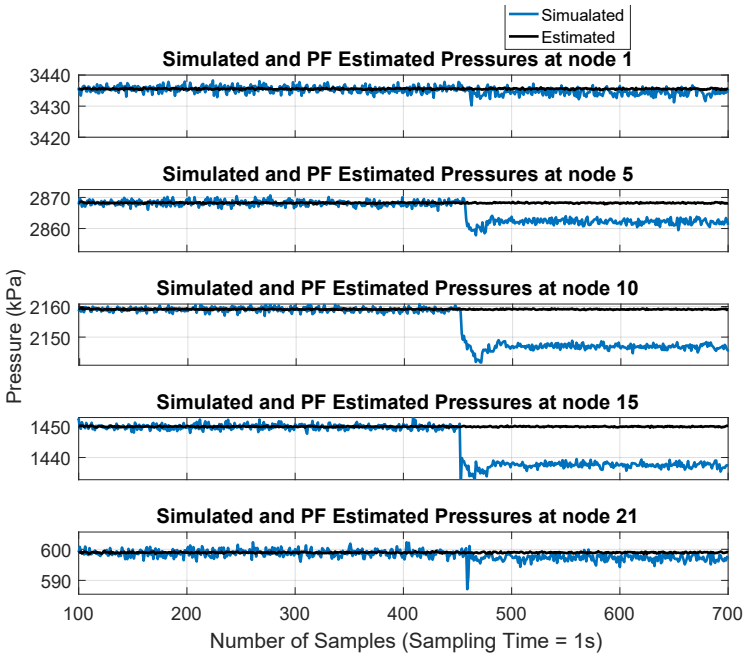


**Figure 3.4:** Residuals between the PF estimated pressures and available true (simulated) pressure measurements for 5% leak at node 13 at 450 s.

## 3.4 Classical Leak Signature

Pipeline leak causes some unique changes in the hydraulics of pipeline. These changes are quite different than normal operating conditions behaviors. These distinct hydraulic behaviors of pipeline leak is termed as 'leak signatures'.

A typical pipeline leak has two important signatures. The first one is the increase in the mass flow-rate of first node and decrease in the mass flow-rate of the last node from the nominal flow-rate (Ostapkowicz, 2014). This is unique because when there is regular step-up in the process, both the first and the last node mass flow-rates will increase accordingly. When there is step-down in the process, both the first and last node mass flow-rates will decrease simultaneously. The phenomena is evident in Figure 3.3 and the bottom part of Figure 3.5.

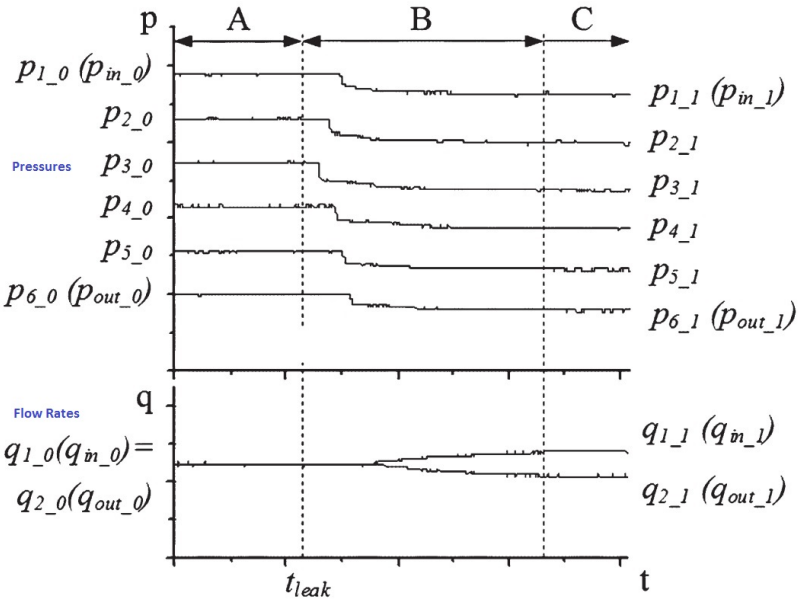


**Figure 3.5:** Leak Signature (Ostapkowicz, 2014). Pressures of all the nodes including the first and last nodes have decreased after the leak initiation time ($t_{leak}$). Mass flow-rate of the first node has increased and last node has decreased after $t_{leak}$.

When a leak occurs, fluid is coming out of pipeline so the decrease in flow-rate at the last node is quite intuitive. To explain the reason behind the increase in the flow-rate of first of node the second leak signature needs to be introduced first. The second signature is the decrease in pressure in all the nodes. Figure 3.4 and the upper part of Figure3.5 show this effect.

This second leak signature can also be attributed to the loss of fluid from the pipeline. Since fluid is coming out from a pressurized pipeline; pressure will also be released along

with the fluid loss. Since the pressure is decreased along the pipeline, back pressure to the first node is also decreased. Hence, resistance to the flow is decreased which ultimately causes the increase in the first node flow-rate. Thus, the hydraulic behavior of the pipeline during a leak is unique.

## 3.5 Concluding Remarks

This chapter gives a brief idea about the existing pipeline leak detection methods. The models for simulating a pipeline flow and the leak were also described from the first principle pipeline hydraulic models. Leak signature was also addressed figuratively.

# Chapter 4

# Model Based Leak Detection Approach using Particle Filter

Existing data-driven (statistical analysis) methods suffer from poor performance during transient conditions. A real-time transient model (RTTM) based method can improve the leak detection performance during a process transition. In this chapter, a model-based leak detection method has been proposed. To handle the uncertainties involved with the model, Monte Carlo simulation based particle filter algorithms (Arulampalam et al., 2002; Ristic et al., 2004) are used to estimate the unmeasured states and refine the estimated values using the available measurements. The residual errors between the model estimates and true measurements are used to detect a leak. In a previous study, a particle filter algorithm was used to detect leaks in a gas pipeline in a simulation environment (Liu et al., 2005).

This study focuses on both gas and liquid transport pipeline. The efficacy of the proposed method is shown via simulation results for leak detection and localization. Fluid leak is detected by comparing the particle filter estimated states (pressure) with the available intermediate pressure measurements. Before discussing the proposed model-based method, the principle of particle filter is discussed in the next section.

## 4.1 Principle of Particle Filters

Particle filter (PF) is a class of non-linear state estimator which uses Monte Carlo simulation. This algorithm tries to approximate the optimal Bayesian solution for a non-linear Bayesian state estimation problem (Arulampalam et al., 2002; Ristic et al., 2004). Before discussing the Particle Filter algorithm, Bayesian state estimation problem is discussed first in the following sub-section.

### 4.1.1 Bayesian State Estimator

The following non-linear stochastic system model is considered to define the non-linear Bayesian state estimation problem -

$$x_k = f_k(x_{k-1}, u_{k-1}, v_{k-1}) \tag{4.1}$$

$$z_k = h_k(x_k, n_k) \tag{4.2}$$

where, equation 4.1 is the state transition equation and equation 4.2 is the measurement equation. Different terms of the above equations have the following interpretations -

- $x_k$ = estimated state vector at $k^{th}$ time instant,
- $f_k$ represents the non-linear state transition equation,
- $h_k$ represents the non-linear measurement equation,
- $u_{k-1}$ represent the input vector at $(k-1)^{th}$ time instant which combines all the known inputs to the process,
- $v_{k-1}$ is the process noise which also captures the modeling error and disturbances at $(k-1)^{th}$ time instant,
- $n_k$ is the measurement noise at $k^{th}$ time instant.

The particle filter or any other state estimators try to find out the optimal estimates of state vector on the condition that all the measurements are available. This condition can be expressed mathematically with the help of conditional probability density as shown in equation 4.3.

$$p(x_k|z_{1:k}) = \frac{p(z_{1:k}, x_k)}{p(z_{1:k})} \tag{4.3}$$

where,

- $z_{1:k}$ represents the set comprising all the measurements until $k^{th}$ time instant,
- $p(x_k|z_{1:k})$ denotes the conditional probability of state vector $x_k$, which is generally known as the posterior density of the state at $k^{th}$ time instant.

Since, measurement $z_k$ only depends on $x_k$ (equation 4.2), hence it is a 'memoryless' relation between $z_k$ and $x_k$ (Balakrishnapillai Chitralekha, 2011), utilizing the definition of conditional probability, the numerator of equation 4.3 can be further decomposed as following -

$$p(z_{1:k}, x_k) = p(z_k|x_k)p(z_{1:k-1}, x_k)$$

Thus, equation 4.1 now becomes,

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(z_{1:k-1}, x_k)}{p(z_{1:k})} \tag{4.4}$$

Similarly, applying the definition of conditional probability to $p(z_{1:k-1}, x_k)$ and $z_{1:k}$, equation 4.4 now becomes,

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})p(z_{1:k-1})}{p(z_k|z_{1:k-1})p(z_{1:k-1})}$$
$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{4.5}$$

where,

- $p(z_k|x_k)$ is the likelihood of the measurement data at $k$th time instant,
- $p(x_k|z_{1:k-1})$ is the prior density of the state at $k$th time instant, which is dependent on the availability of all previous measurements,
- $p(z_k|z_{1:k-1})$ is a normalizing constant and independent of the state.

The likelihood represents a conditional probability such that the measurement ($z_k$) at the current ($k$th) time instant is conditioned on the current value of the state ($x_k$). The stochastic relationship between $z_k$ and $x_k$ given by equation 4.2 is characterized by this conditional density term. Using the probability density function of the noise in the measurement, this likelihood can be evaluated as shown in equation 4.6 (Balakrishnapillai Chitralekha, 2011).

$$p(z_k|x_k) = \int \delta(z_k - h_k(x_k, n_k))p(n_k)dn_k \tag{4.6}$$

where, $\delta(.)$ is the Dirac-delta function. The prior density term $p(x_k|z_{1:k-1})$ is not readily available. If all the measurement till the previous time instant $(k-1)$ is available, then it can be assumed that the prior density is somehow known. The Chapman-Kolmogorov equation (equation 4.7) gives the recursive relationship to calculate the prior density from posterior density of previous step. This equation uses the Markov property of the system model (equation 4.1).

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{4.7}$$

The term $p(x_k|x_{k-1})$ can be evaluated using the distribution of process noise term as in equation 4.8

$$p(x_k|x_{k-1}) = \int \delta(x_k - f_k(x_{k-1}, u_{k-1}, v_{k-1}))p(v_{k-1})dv_{k-1} \tag{4.8}$$

Now, the task of propagation can be divided into two step:

- Prediction Step: The Chapman Kalmogorov equation gives the prior density of states which ultimately forms the prediction step.

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{4.9}$$

- Correction Step: The posterior density evaluated by equation 4.5 is the correction or update step. The likelihood ($p(z_k|x_k)$) of equation 4.5 is calculated using equation 4.6.

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{4.10}$$

Equation 4.9 and 4.10 are general steps of any state estimator (Balakrishnapillai Chitralekha, 2011). In the following section, the particle filter algorithm will be discussed.

### 4.1.2 The Sequential Importance Sampling (SIS) Algorithm

The particle filter is a Monte Carlo method based algorithm . The basic idea of Monte Carlo method is to visualize a probability distribution as a function or plot of density function with the random variable as independent variable. The term 'conditional' denotes that the available measurements can influence the shape of this plot (Balakrishnapillai Chitralekha, 2011). The particle filtering technique can be called a recursive Bayesian filter by Monte Carlo simulations. The posterior distribution, $p(x_k|z_{1:k})$, can be approximated by a set of random samples with associated weights. The estimates of the states can be obtained by using these samples and weight. With increase of the number of samples, this approximation becomes an equivalent of the posterior distribution and the particle filter approaches the optimal Bayesian estimate (Arulampalam et al., 2002; Ristic et al., 2004). That is why particle filters are known as sub-optimal filter.

Among several variants of particle filtering approaches, the most basic one is the sequential importance sampling (SIS) algorithm. The details of the SIS algorithm are discussed as follows.

Let $\{x^i_{0:k}, w^i_k\}_{i=1}^{N_S}$ represents a random variable which characterizes posterior distribution $p(x_{0:k}|z_{1:k}$, where

- $\{x^i_{0:k}, i = 0, ....., N_S\}$ is a set of random points or samples,
- $\{w^i_k, i = 0, ....., N_S\}$ are the associated weights of the random points,
- $x_{0:k} = \{x_j, j = 0, ....., k\}$ is the set of all states at time instant $k$

If the weights are normalized such that $\sum_i w_k^i = 1$, the posterior distribution can be approximated with a discrete weighted form as in equation 4.11.

$$p(x_{0:k}|z_{1:k}) \approx \sum_{i=1}^{N_S} w_k^i \delta(x_{0:k} - x_{0:k}^i) \tag{4.11}$$

The weights are chosen using the Importance Sampling principle (Arulampalam et al., 2002) as discussed below.

Suppose, $p(y) \propto \Pi(y)$ is such a probability density from which it is difficult to draw samples but $\Pi(y)$ can be evaluated. Again, $y^i \sim q(y), i = 1, ...., N_S$ be samples that are easily generated from a proposal $q(.)$ which is known as *importance density*. Thus, the density function $p(y)$ can be approximated as in equation 4.12 which is a weighted approximation.

$$p(y) \approx \sum_{i=1}^{N_S} w_k^i \delta(y - y^i) \tag{4.12}$$

where, $w^i$ is the normalized with of the $i$-th particle or sample or point and defined as in equation 4.13.

$$w^i \propto \frac{\Pi(y^i)}{q(y^i)} \tag{4.13}$$

Hence, if the samples, $x_{0:k}^i$, are drawn from a similar importance density such as $q(x_{0:k}|z_{1:k})$, the normalized weight in equation 4.11 can be defined as in equation 4.14 (Arulampalam et al., 2002; Ristic et al., 2004).

$$w_k^i \propto \frac{p(x_{0:k}^i|z_{1:k})}{q(x_{0:k}^i|z_{1:k})} \tag{4.14}$$

Now at each iteration, one will want to approximate $p(x_{0:k}|z_{1:k})$ with the samples available for approximating $p(x_{0:k-1}|z_{1:k-1})$. If the importance density can be factorized as

$$q(x_{0:k}|z_{1:k}) = q(x_k|x_{0:k-1}, z_{1:k})q(x_{0:k-1}|z_{1:k-1}) \tag{4.15}$$

then one can obtain samples $x_{0:k}^i \sim q(x_{0:k}|z_{1:k})$ using each of the existing samples $x_{0:k-1}^i \sim q(x_{0:k-1}|z_{1:k-1})$ with the new state $x_k^i \sim q(x_k|x_{0:k-1}, z_{1:k})$. Now,

$$\begin{aligned}
p(x_{0:k}|z_{1:k}) &= \frac{p(z_k|x_{0:k}, z_{1:k-1})p(x_{0:k}|z_{1:k-1})}{p(z_k|z_{1:k-1})} \\
\Rightarrow p(x_{0:k}|z_{1:k}) &= \frac{p(z_k|x_{0:k}, z_{1:k-1})p(x_k|x_{0:k-1}, z_{1:k-1})}{p(z_k|z_{1:k-1})} p(x_{0:k-1}|z_{1:k-1}) \\
\Rightarrow p(x_{0:k}|z_{1:k}) &= \frac{p(z_k|x_k)p(x_k|x_{k-1})}{p(z_k|z_{1:k-1})} p(x_{0:k-1}|z_{1:k-1}) \\
\Rightarrow p(x_{0:k}|z_{1:k}) &\propto p(z_k|x_k)p(x_k|x_{k-1})p(x_{0:k-1}|z_{1:k-1})
\end{aligned} \tag{4.16}$$

The weight update equation can be updated by substituting equations 4.15 and 4.16 into equation 4.14.

$$w^i \propto \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)p(x_{0:k-1}^i|z_{1:k-1})}{q(x_k^i|x_{0:k-1}^i, z_{1:k})q(x_{0:k-1}^i|z_{1:k-1})}$$
$$\rightarrow w_k^i = w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{0:k-1}^i, z_{1:k})} \tag{4.17}$$

Moreover, if $q(x_k|x_{0:k-1}, z_{1:k}) = q(x_k|x_{k-1}, z_{1:k})$, the importance density becomes dependent only on $x_{k-1}$ and $z_k$. This is a common case when only a filtered estimate of $p(x_k|z_{1:k})$ is required at every time step. Under this condition, only $x_k^i$ is stored and $x_{0:k-1}^1$ and $z_{0:k-1}$ are discarded. The modified weight update equation has the following form in that case -

$$w_k^i \propto w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)} \tag{4.18}$$

The corresponding posterior density can be approximated by equation 4.19.

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_S} w_k^i \delta(x_{0:k} - x_k^i) \tag{4.19}$$

The weight is calculated using equation 4.18. When $N_S \rightarrow \infty$, the approximation of equation 4.19 becomes equivalent to the true posterior density function $p(x_k|z_{1:k})$. A pseudo-code of the SIS algorithm is given by Algorithm 1.

**Algorithm 1** : SIS Particle Filter

1: $[\{x_k^i, w_k^i\}_{i=1}^{N_S}] = SIS[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_S}, z_k]$
2: **for** $i = 1 : N_S$ **do**
3: Draw $x_k^i \sim q(x_k|x_{k-1}^i, z_k)$
4: Assign the particle a weight, $w_k^i$ according to equation 4.18
5: **end for**

### The Degeneracy Problem of SIS Algorithm

A common phenomenon of SIS algorithm is that after few iterations, weights of all the particles become negligible except one. This is called the so called 'degeneracy problem'. Since, the variance of the importance weights can only increase over time, the degeneracy problem is obvious (Arulampalam et al., 2002). So, a large computational effort is devoted in updating such particles whose impact on the approximation of $p(x_k|z_{1:k})$ is negligible. A measure of the degeneracy can be obtained by effective sample size ($N_{eff}$) as defined in equation 4.20 (Arulampalam et al., 2002).

$$N_{eff} = \frac{N_S}{1 + \text{Var}(w_k^{*i})} \tag{4.20}$$

where, $w_k^{*i}$ is defined by equation 4.21. It is referred to as the 'true weight'.

$$w_k^{*i} = \frac{p(x_k^i|z_{1:k})}{q(x_k^i|x_{k-1}^i, z_k)} \tag{4.21}$$

Though the true weight cannot be obtained exactly, but an approximation of the effective sample size ($\widehat{N_{eff}}$) can be made using the equation 4.22. The term $w_k^i$ used in this equation, is the normalized weight which can be obtained from equation 4.17.

$$\widehat{N_{eff}} = \frac{1}{\sum_{i=1}^{N_S}(w_k^i)^2} \tag{4.22}$$

It is noteworthy that the value of $\widehat{N_{eff}} \leq N_S$. If $\widehat{N_{eff}}$ is much smaller than $N_S$, the degeneracy problem is serious. One solution to the problem is to use a very large $N_S$, but this will increase the computational effort. The practical way to handle the problem is using either the resampling algorithm or using an effective importance density. Both of these methods are described in the Appendix C.

### 4.1.3 Sampling Importance Resampling (SIR) Particle Filter

To handle the degeneracy problem of the SIS particle filter, the SIR particle filter is proposed. This variant of particle filter uses a sub-optimal choice of the importance density function along with the resampling algorithm to tackle the degeneracy problem.

The most popular sub-optimal choice of importance density is the transitional prior density as given in equation 4.23.

$$q(x_k|x_{k-1}^i, z_k) = p(x_k|x_{k-1}^i) \tag{4.23}$$

Hence, combining equations 4.23 and 4.18 gives

$$w_k^i \propto w_{k-1}^i p(z_k|x_k^i) \tag{4.24}$$

**Description of SIR Particle Filter** The class of particle filter that uses the transitional prior as importance density is known as the 'sampling importance resampling' or SIR particle filter. Algorithm 2 shows the psuedocode for SIR particle filter (Arulampalam et al., 2002; Ristic et al., 2004). Unlike the optimal importance density, the sub-optimal importance density cannot be computed before the particle are propagated to time $k$. Moreover, if the transitional prior density is a much broader distribution than the likelihood ($p(z_k|x_k)$), then only one particle will be assigned a high weight. The consequence is that the particles will degenerate rapidly and the particle filter will not work. Additionally, since SIR filter

**Algorithm 2** : SIR Particle Filter (Ristic et al., 2004)

1: $[\{x_k^i, w_k^i\}_{i=1}^{N_S}] = SIR[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_S}, z_k]$
2: **for** $i = 1 : N_S$ **do**
3: Draw $x_k^i \sim p(x_k|x_{k-1}^i)$
4: Calculate $w_k^i = p(z_k|x_k^i)$
5: **end for**
6: Calculate total weight: $t = SUM[\{w_k^i\}_{i=1}^{N_S}]$
7: **for** $i = 1 : N_S$ **do**
8: Normalize: $w_k^i = t^{-1} w_k^i$
9: **end for**
10: Resample using algorithm 3 : $[\{x_k^i, w_k^i, -\}_{i=1}^{N_S}] = RESAMPLE[\{x_k^i, w_k^i\}_{i=1}^{N_S}]$

is independent of measurement ($z_k$), the filter can be inefficient and sensitive to outliers. The remedy is to use auxiliary SIR particle filter (Arulampalam et al., 2002; Ristic et al., 2004)).

However, the advantage of using the SIR particle filter is that the importance weights can be easily calculated and importance density can be easily sampled (Ristic et al., 2004)). Due to these advantages, the SIR particle filter was used in this thesis.

### 4.1.4 Example to Demonstrate the Efficacy of Particle Filtering

#### Single Tank System

The particle filter was first tested with a simple single tank system as shown in Figure 4.1. The model of the system is given by equation 4.25.

$$\frac{dh}{dt} = \frac{q_i}{A} - \frac{C_v}{A}\sqrt{h} \tag{4.25}$$

where,

- $h$ = tank height (the only state in this case),
- the limits for $h$ are, $h_{max}$ = 100 cm and $h_{min}$ = 0 cm
- tank cross-sectional area, $A = 0.02m^2$,
- valve co-efficient, $C_v = 0.5$,
- tank inlet flow, $q_i$ is the disturbance variable since there is no control over this variable,
- tank outlet flow, $q_o$ is dependent on tank height as $C_v\sqrt{h}$

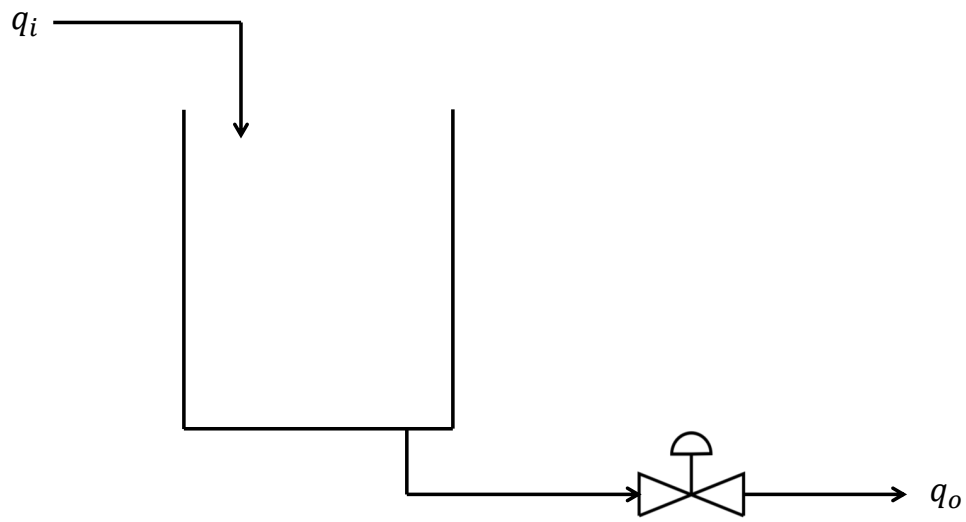


**Figure 4.1:** Schematic diagram of a simple single tank system.

In this example $h$ is the measurement, there is no unknown state to estimate. The particle filter acts as a noise filter in this case. It was assumed that the measurement noise variance is 2 and process noise variance is 0.25. In this simulation, number of particles used was 50.

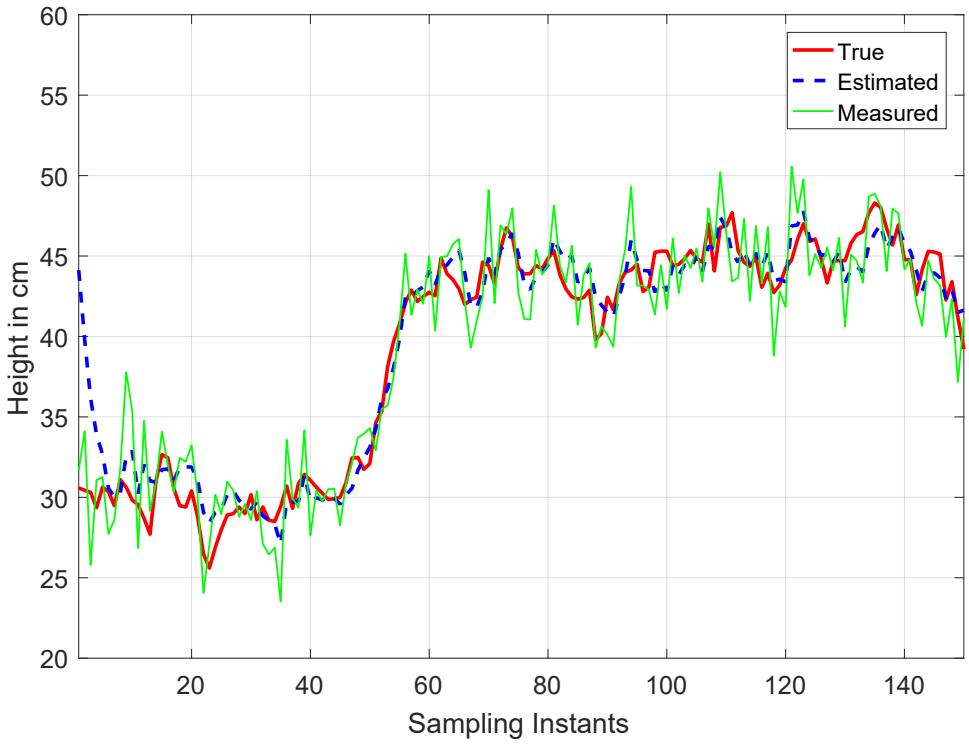


**Figure 4.2:** Performance of particle filter (PF) as a noise filter.

From Figure 4.2, it is clear that the particle filter was able filter out noise effect of the measurement to estimate the true tank height quite efficiently. It is noteworthy that though there was a difference in the initial condition, the PF was able to track the true height. In this example, true height indicates the height calculated by the model in the absence of measurement noise. The measurement noise was incorporated in the measurement. There was a step change in the inlet flow, $q_i$ at sampling instant 50. The PF efficiently tracked the step change too.

**An arbitrary non-linear system with two unknown states**

Let the following be a non-linear stochastic model with two unknown states -

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + v_k \\ y_{k+1} &= Cx_{k+1} + n_{k+1} \end{aligned} \tag{4.26}$$

where,

- $x_k$ is the state vector at $k$ th time instant and

$$x_k = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- $u_k$ is the input to the system is denoted by vector $u^1$ for the $1^{st}$ 100 samples and $u^2$ for the $2^{nd}$ 100 samples to introduce a step change in the system. The vectors $u^1$ and $u^2$ are as follows -

$$u^1 = \begin{bmatrix} 3 \\ 0.1 \end{bmatrix}$$

$$u^2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

- state transition matrix

$$A = \begin{bmatrix} 0.19 & 0.25 \\ 2 & 0.0004 \end{bmatrix}$$

- input matrix

$$B = \begin{bmatrix} 0.09 & 0.5 \\ 1 & 0.0001 \end{bmatrix}$$

- $v_k$ is process noise at $k$ th time instant

- output vector

$$C = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}$$

- $n_{k+1}$ is the measurement noise at $(k+1)^{th}$ time instant.

Figure 4.3 shows the output of the model. Since the complete model is known, the unknown states were also estimated for verification purpose. However, the PF does not have any clue about these states. The particle filter has only the knowledge of output measurement ($y$). Since in reality, all the systems suffer from process and measurement noise, the Gaussian noise was added with the model estimated states ($x_{1,m}$, $x_{2,m}$) and output ($y_m$). The noisy measurement ($y$) was used in the PF algorithm to estimate the unknown states ($x_{1,est}$, $x_{2,est}$).

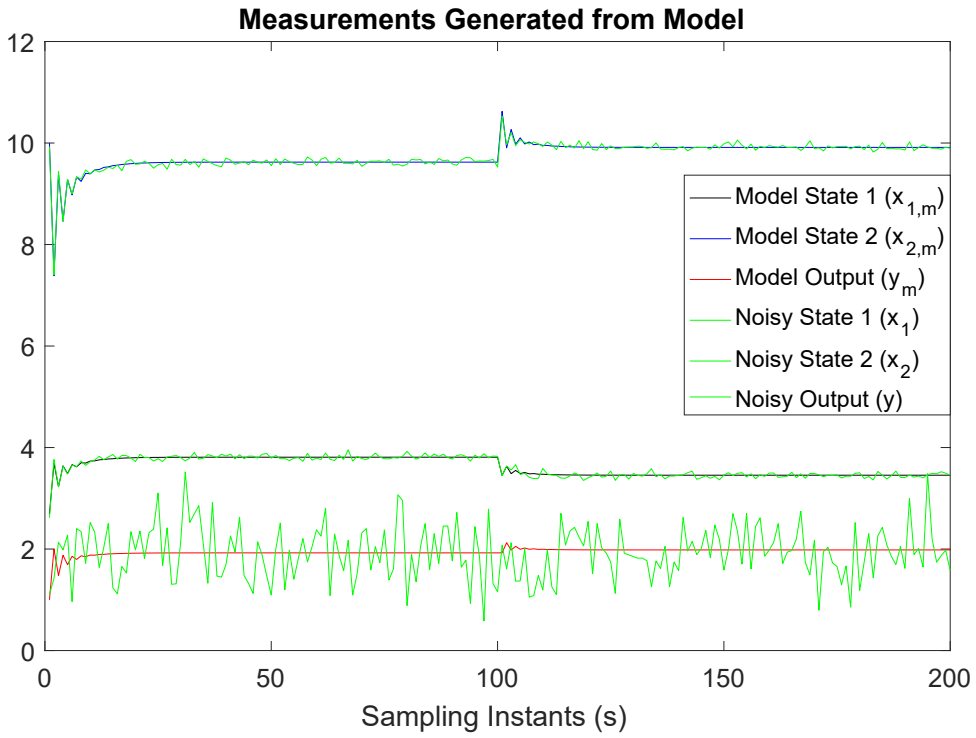


**Figure 4.3:** Data generation from the model. In this case, the model generated states are not available to PF, only the output is available of PF.

Figure 4.4 shows the performance of particle filter on the above-stated model. It is clear that the PF estimated states ($x_{1,est}$, $x_{2,est}$) converged with the unknown noisy states ($x_1$, $x_2$), even though the initial condition for PF estimation was quite different from the unknown states ($x_1$, $x_2$). The PF algorithm was also able to track the step change initiated in the process at $100^{th}$ sample by changing the process input from $u^1$ to $u^2$.

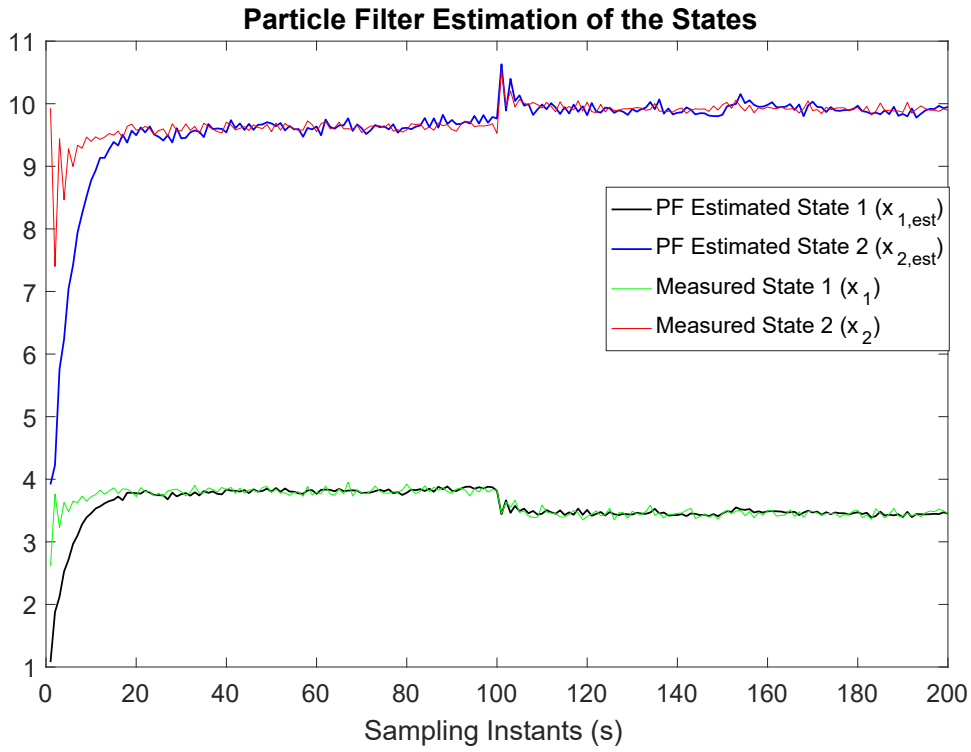


**Figure 4.4:** Performance of particle filter (PF) as a state estimator.

## 4.2 The Proposed Model Based LDS Algorithm

The hydraulic model of pipeline flow (see section 3.2) is the heart of the proposed model-based algorithm. The model estimates are being refined by particle filter which uses some of the available measurements to give more realistic results. Since the leak event is uncertain, the size and location of a leak is unknown and no one knows when a leak will be there, it is difficult to tailor the model for a leak. Rather, the model can estimate the normal (no-leak) behavior of the pipeline under the exposed operating condition. When there is a leak, the true measurements will deviate from this particle filter (PF) estimated values. Calculating the residual between the true measurements and PF estimated values using equation 1.3 and comparing it with the leak signature as described in section 3.4, the leak is detected and localized in the proposed model-based algorithm.

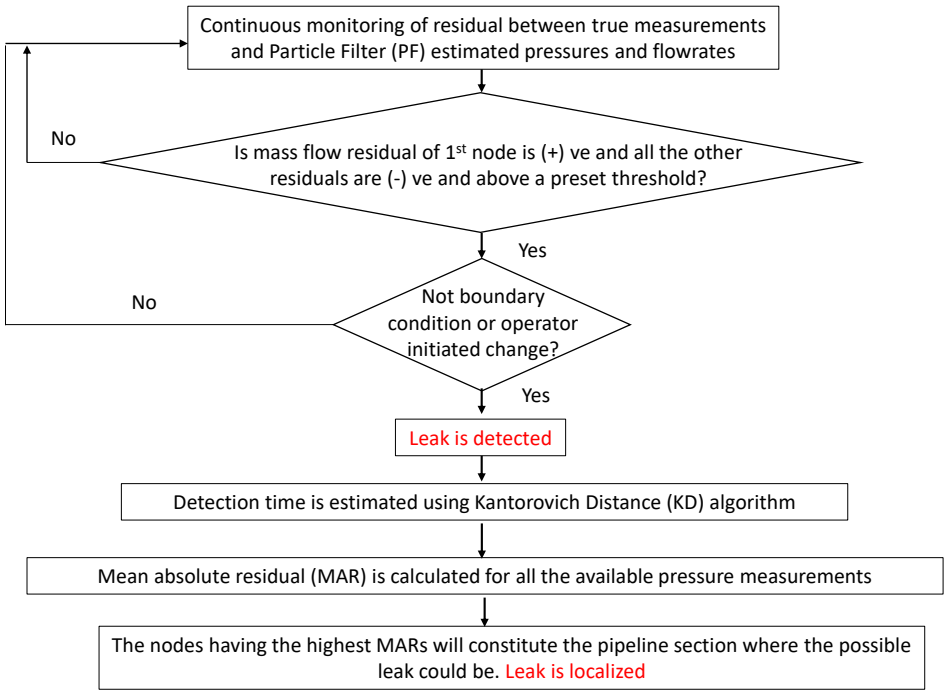


**Figure 4.5:** Flow diagram for the particle filter (PF) based LDS algorithm.

Figure 4.5 shows the flow diagram of the proposed model-based leak detection method. The first step is to monitor the residuals between the available true measurements and their corresponding PF estimated values. If the residual of the flow-rate at the first node is positive and mass flow residual of the last node have negative residuals, this conforms to the leak first signature as discussed at section 3.4. Similarly, if the residuals for all the nodes where real pressure measurements are available have negative residuals, then the second leak signature is also present. Thus, the leak can be detected with the proposed algorithm.

Sometimes the leak signatures can be produced due to noise. To get rid of this issue, thresholds can be preset for these residuals. The choice of these thresholds depends upon the specific process and criteria chosen by the user. A simple way to choose the threshold is to use the $2\sigma$ or $3\sigma$ value, where $\sigma$ refers to the variance of noise corresponding to each of the true measurements. However, if the leak size is smaller than the chosen threshold, that leak cannot be determined.

When the residuals conform to the leak signature then it is further checked that this change was not initiated by the operators or by changing the boundary and process operating conditions. When this is also confirmed, the leak alarm is raised. The next phase is to estimate the change-time or leak-time using the Kantorovich distance (KD) algorithm. The true pressure measurements are used with the KD algorithm to detect the leak-time ($t_{leak}$).

Since the nearest neighboring nodes of the leak point will have the highest effect of the leak (Geiger, 2006), the minimum time required to reach the peak of KD corresponding to available pressure measurements will give a notion of the leak-time ($t_{leak}$). Depending upon the measurement noise, these pressure measurements may need to go through some simple filtering techniques to reduce the effect of noise.

At the next stage mean absolute residual (MAR) is calculated using the estimated leak-time $t_{leak}$. The MAR is calculated using the equation 4.27. To calculate the mean, 150 sample before the leak-time and 150 samples after leak-time was considered. Number of samples before and after the leak-time may vary case to case.

$$\text{MAR} = \frac{\sum_{i=t_{leak}-150}^{t_{leak}+150} \text{abs}(p_{i,j}^{t} - p_{i,j})}{\sum i} \tag{4.27}$$

where,

- $t_{leak}$ is the leak-time estimated using the KD algorithm,
- $p_{i,j}^{t}$ is the true pressure measurement available at node $j$,
- $p_{i,j}$ is the PF estimated pressure at node $j$.

The leak is localized by isolating a smaller section of the pipeline whose corresponding nodes have the highest MARs. So, the proposed model-based method can detect and localize leak, though it cannot pinpoint the exact location of the leak, it can isolate a smaller section of the pipeline which reduces a lot of effort to identify leak location by not-technical methods.

## 4.3 Detection and Localization of Leaks using the Proposed Model Based LDS Method

### 4.3.1 Simulation Preparation

In this study, the simulated pipeline is 40 km in length with 0.4 m inlet diameter. For simulation purposes, the pipeline was divided into 20 equal sections (21 nodes). Therefore, each section is 2 km in length and the total number of states are 42 including pressure and mass flow-rate measurements at each of these nodes. In this case, density is considered as constant, which is reasonable in the sense that the variation of density with the operational temperature difference is less than 1%. In Al-Khomairi (1995) it was also shown that choosing a constant density is a valid option. In this study, Naptha was chosen as the working fluid for these simulations. So, properties of Naptha were used in the corresponding equations. Total simulation length was 700 s with sampling-time as 1 s.

Here, the two fixed boundary conditions were the pump just before node 1 and the valve just after at node 21 (Figure 2.10) to control the pipeline flow conditions. The boundary conditions should be such that those are free from the effects of the process faults in the process and they can be used to change the process operating conditions. In that sense by changing the speed of pump at the inlet of pipe or changing the valve opening at the outlet of the pipe are two reasonable choices for boundary conditions. Other fixed boundary conditions can also be used. The available measurements are assumed as two mass flow-rate measurements at node 1 ($W_{1,j}$) and node 21 ($W_{21,j}$) and five pressure measurements available at nodes 1 ($p_{1,j}$), 5 ($p_{5,j}$), 10 ($p_{10,j}$), 15 ($p_{15,j}$) and 21 ($p_{21,j}$).

### 4.3.2 Validating the PF Model

Before applying the PF based LDS algorithm, it is necessary to check the performance of the particle filter in normal condition. If the particle filter can estimate the process in normal condition, only then the proposed model-based method can work efficiently. Figure 4.6 shows that the PF can track the pipeline behavior well even though there was a significant difference in the initial condition. *In all these results, 'simulated' means the simulated pipeline data and 'estimated' indicates particle filter estimated values.*

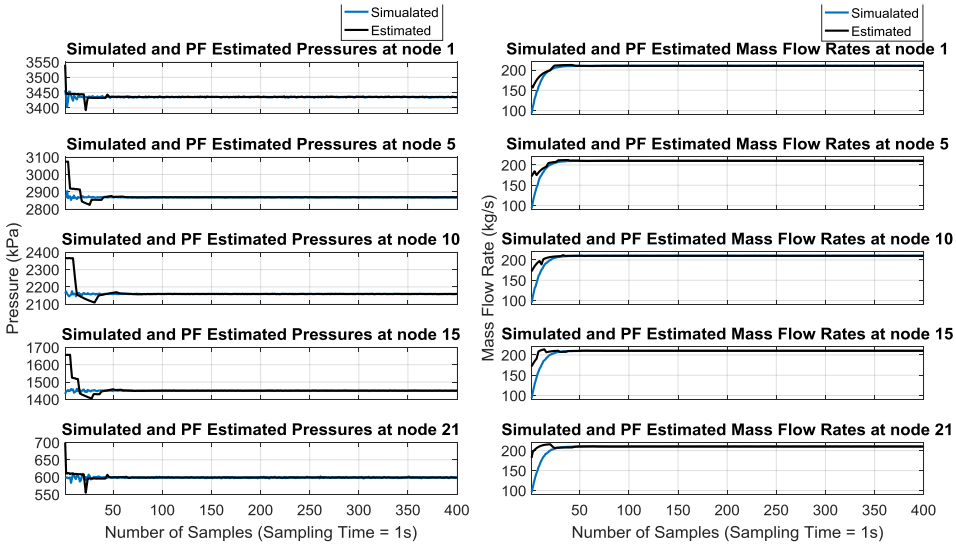


**Figure 4.6:** Simulated and particle filter estimated pressures flow-rates for nodes 1, 5, 10, 15 and 21. The left panel shows the pressure measurement and the right panel shows the mass flow-rates.

Figure 4.7 is a zoomed-in version of Figure 4.6, which indicates that the PF is not only tracking the pipeline behavior but also filtering out the effect of the noise. These two figures have clearly depicted the efficacy of PF. One crucial choice in the particle filter is to select

the optimal number of particles. Generally, the more the number of particles used, the better the accuracy of estimation but this also increases the computational effort (Modisette et al., 2013). All the simulations of this study were done by taking 50 particles. In this case, mass flow-rates of the first ($W_{1,j}$) and last nodes ($W_{21,j}$) including pressure measurement of node 11 ($p_{11,j}$) were used as the measurements for PF to refine the model estimates. The next section discusses the leak detection and localization results of this simulated case.

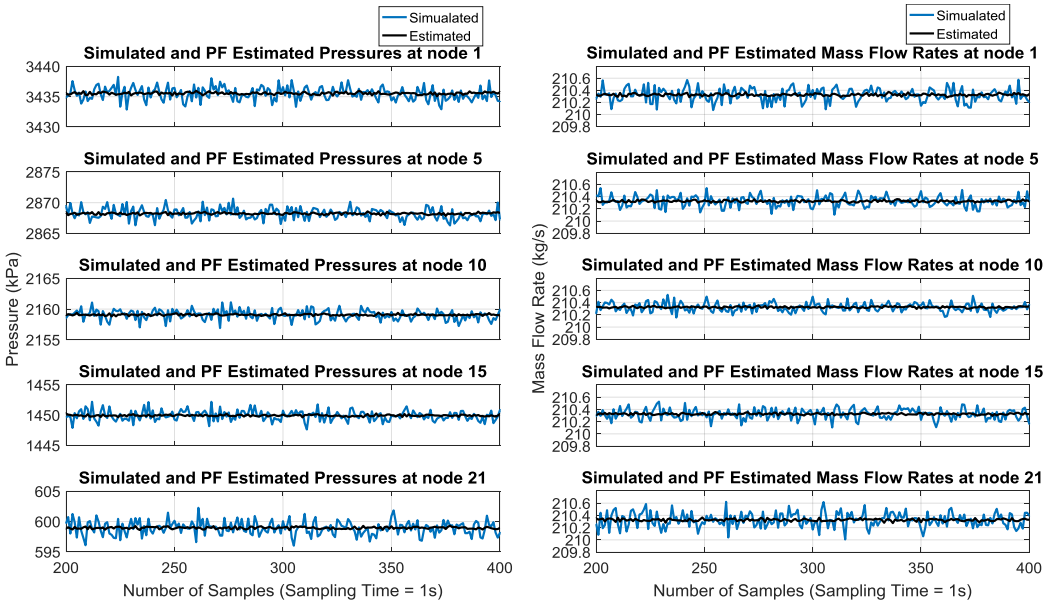


**Figure 4.7:** Zoomed-in version (from 200 s to 400 s) of Figure 4.6. The left panel shows the pressure measurement and the right panel shows the mass flow-rates.

### 4.3.3 Simulated Leak Detection and Localization

The basic idea of the proposed leak detection and localization method is based on the deviation method. A leak was not introduced in the particle filter model. So, the particle filter model can generate the behavior of the pipeline under normal operating conditions. Whereas, due to leak the behavior of the simulated pipeline system will change. In this section three (3) simulated leak case studies are discussed. They are -

- 5% leak at node 6 at 450 s
- 3% leak at node 9 at 450 s
- 5% leak at node 13 at 450 s

In this section, the efficacy of the proposed model-based algorithm is shown with the results from the leak at node 13. Results of the rest two simulated leak case studies are given in Appendix D.

Figures 3.4 and 3.3 show the residual between the true pressure and flow-rate measurements and PF estimated values. In this section, true measurements represent the measurements generated using the simulated pipeline. It is clear from Figure 3.4 that true pressure is decreasing at all the nodes after the leak has been initiated at node 13. Figure 3.3 shows that the upstream nodes of the leak node (node 13) have higher true flow-rates than the estimated flow-rates. Similarly, the downstream nodes have true flow-rates lower than the estimated flow-rates. Hence, this phenomenon matches with the leak signatures. Obviously, the residuals have to pass the preset threshold conditions.

At the next stage, the KD algorithm was applied to the available pressure measurements to detect the leak-time. In this case, the leak-time ($t_{leak}$) was detected as 474 s. This delay in detection is because calculation of online KD score starts only after the accumulation of $2m + k - 1$ samples. Figure 4.8 shows the calculation of KD scores.

After leak-time ($t_{leak}$) is estimated, the mean absolute residual for each of the available pressure measurements are calculated using equation 4.27. These values are then used to localize the leak. The upper panel of Figure 4.9 shows that node 10 and node 15 have two highest MARs. Hence, it can be concluded that the leak is in between node 10 and node 15. Since node 15 has the higher residual than that of node 10, so it can be further predicted that the leak is nearer to node 15 than node 10. Thus physical search can be started from node 15 to find the exact leak location faster.

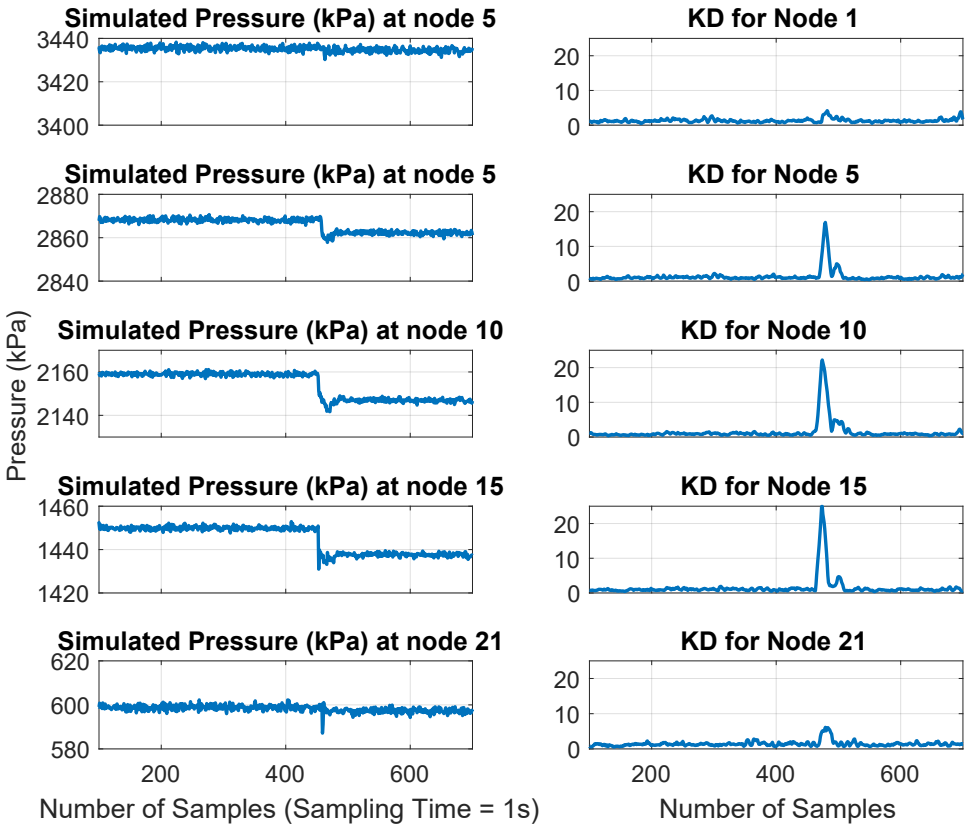


**Figure 4.8:** Detecting change time point using Kantorovich distance algorithm on the available true (simulated) pressure measurements for 5% leak at node 13 at 450 s.

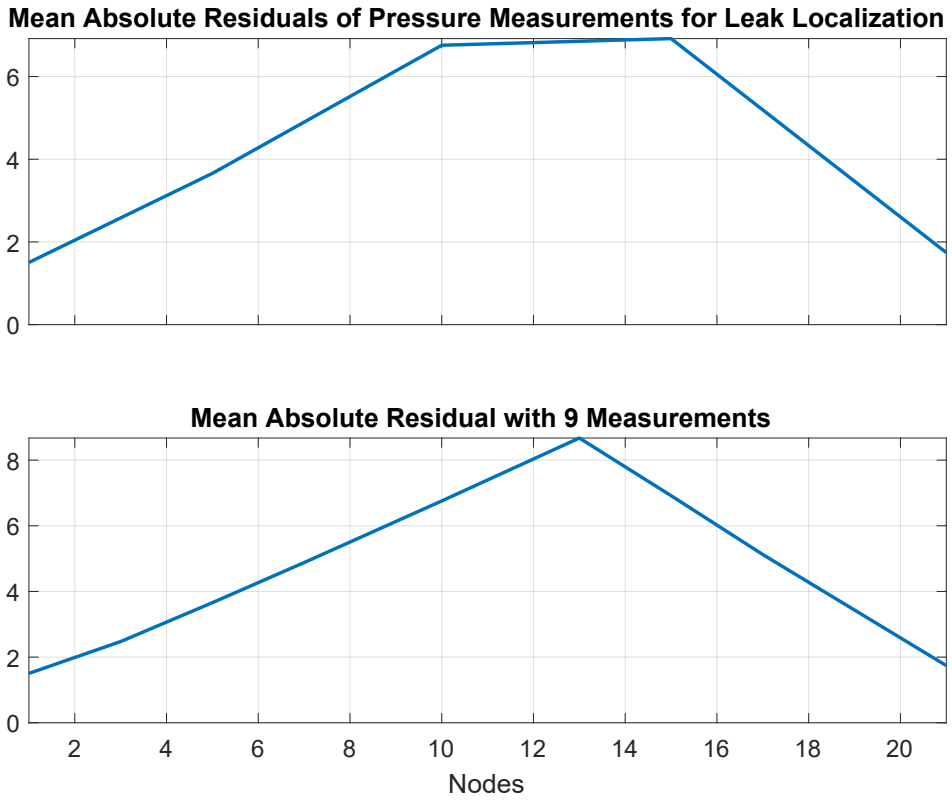


**Figure 4.9:** Leak localization using the mean absolute residual calculated around detected change time point for 5% leak at node 13 at 450 s.

The bottom panel of Figure 4.9, indicates the leak localization efficiency would be better if more intermediate pressure measurements were available. This panel was generated with 9 available pressure measurements. Previously, in Figure 3.4 pressure measurements of nodes 1, 5, 10, 15 and 21 were shown. The additional four pressure measurements are at nodes 3, 7, 13 and 17 as shown in Figure 4.10. Thus, the bottom panel of this figure pinpoints the leak location at node 13.

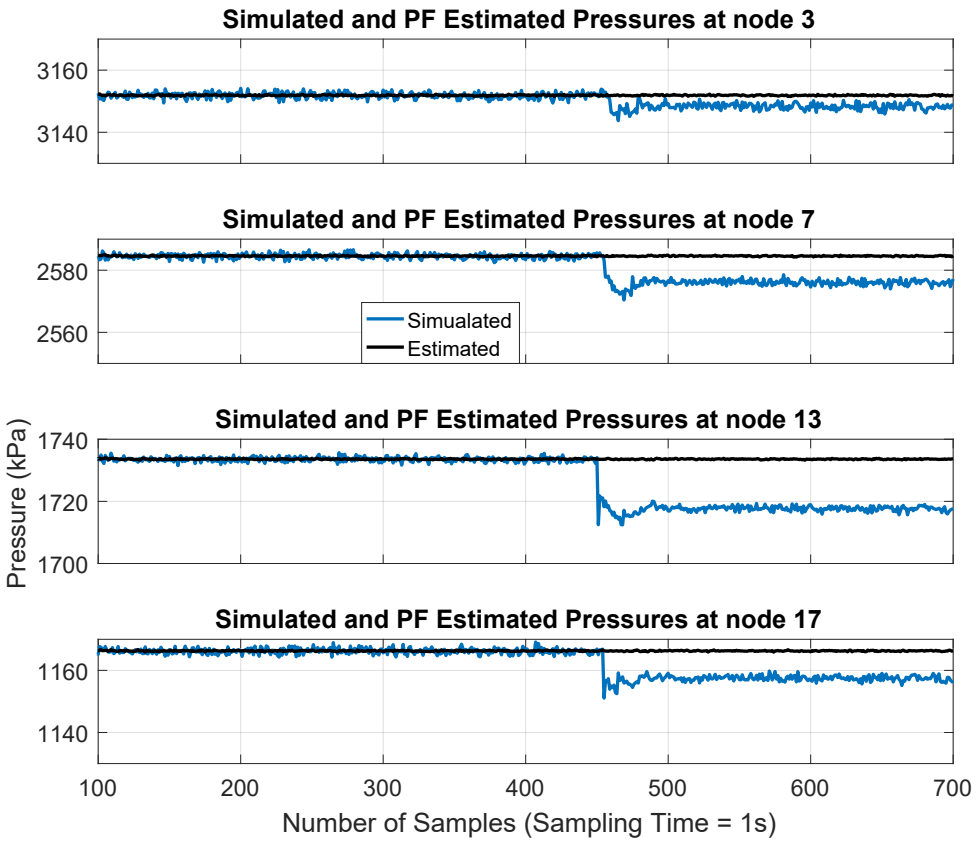


**Figure 4.10:** Residuals between the PF estimated pressures and additional available true (simulated) pressure measurements at nodes 3, 7, 13 and 17 for 5% leak at node 13 at 450 s.

### 4.3.4 Challenges with Real Data

The particle filter based leak detection method was tried to apply to data from a real pipeline. Unfortunately, the method did not work well with the real data. The possible reasons behind this failure are as given :

#### Uncertainty in the Model

A good model is the primary requirement for any model-based algorithm. If the model parameters or structure are uncertain, the residual generated with the model will not be able to capture the actual fault. In this case, the obtained real data was for an old pipeline. The pipeline parameters such :

- pipeline diameter may have been changed due to scaling or damaging of the pipeline.
- pipeline elastic property may have been changed.

All these have effect on the sonic velocity of the fluid through the pipeline. Moreover, the fluid viscosity data was not available in this case. So, the frictional loss in the pipeline was uncertain. Trial and error method was applied, but the model did not converge with the real data even in the normal condition (leak-free).

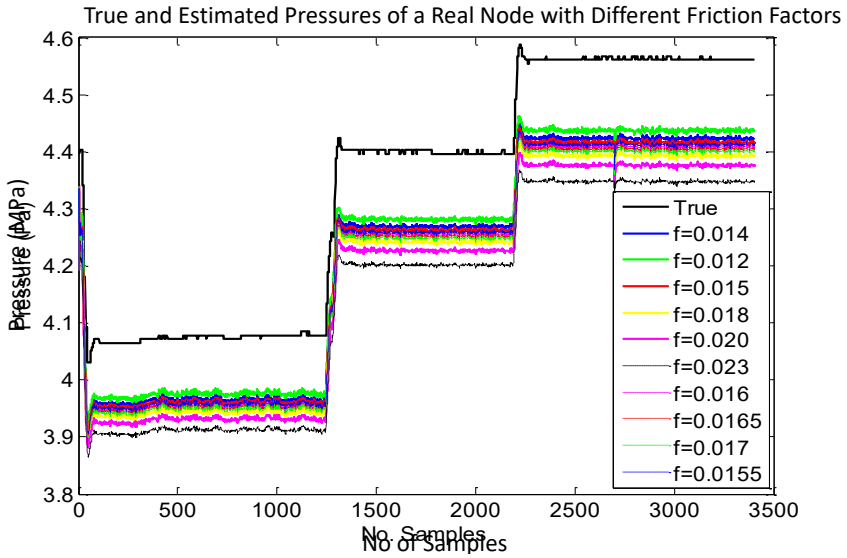


**Figure 4.11:** True and estimated pressures at a real node with different values of friction factor.

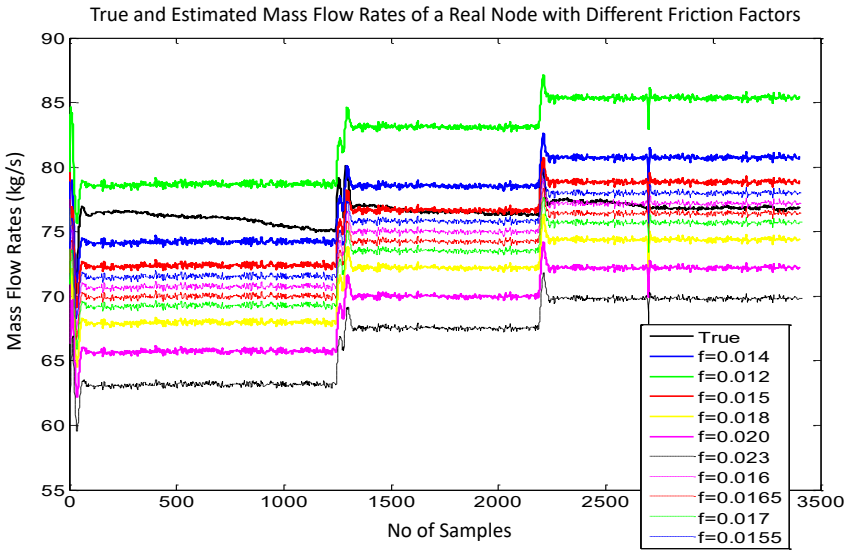


**Figure 4.12:** True and estimated mass flow-rates at a real node with different values of friction factor.

Figures 4.11 and 4.12 are two such examples where the parameter friction factor was

tried with different values. The theoretical calculation identifies the value of friction factor should be as 0.014. But both the pressure and mass flow-rates predicted by the model deviated from the true measurement with a good amount. Moreover, changing the friction factor though improves mass flow-rate significantly, but improvement in the pressure measurement was negligible. Additionally, amount of bias is not constant, which made the tuning process difficult.

**Scarcity of Measurements**

The particle filter was used as state estimator along with the model to help refine the estimated values of the model. Since the model in this case was uncertain, if there were sufficient amount of reliable measurements available, the particle filter might get a closure match to the real data. Again, it is to be reminded, the state estimator cannot improve the quality of the estimates significantly if the model uncertainty is too high.

There were only 5 pressure measurements and 2 end node flow measurements were available for the 51 km long pipeline. The first three pressure measurements were within the first 5 km of the pipeline. The last two pressure measurements were in the last 11 km. So, the middle 35 km there was no pressure measurements. So, there was a large uncertainty in this section. Among these five pressure measurements, node 1 and node 5 pressures were chosen as boundary conditions since actual boundary conditions were not available in this case. Even the parameter estimation algorithms did not work well to find a good estimate of the pipeline parameters.

## 4.4 Concluding Remarks

This chapter includes the study of the application of the particle filter for pipeline leak detection. Simulation results show that the particle filter works very well to capture the dynamics of the pipeline provided that the model is good and hence it can serve the purpose of a soft-sensor.

The proposed leak detection system can detect leaks quite efficiently. Although leak localization is not yet exact, it can at least isolate the smallest possible section of the pipeline which is suspected to have a leak which is a great improvement over the existing LDS. However, due to uncertainties in the model parameters and scarcity of measurement, this model-based method did not work well for the real data.

# Chapter 5

# Data Driven Leak Detection Approach

To avoid the problems of the model-based leak detection approach to real data, a novel data-driven leak detection method is proposed in this chapter. This approach is based the Kantorovich distance algorithm as described in section 2.2. The proposed data-driven LDS method was successfully implemented on both simulated leak in transient pipeline condition and on industrial pipeline in real-time during the annual regulatory leak tests of Suncor pipelines. The real-time test leaks were generated by withdrawing fluid at a specific location of the industrial pipeline and storing it to a temporary reservoir. The withdrawn fluid was pumped back to the pipeline after the test was completed. Two real-time test leaks were generated with the size of the small leak as 1% of the nominal flow-rate and the size of the large leak was 7%. In both the cases the proposed data driven LDS was successful in detecting and localizing the leaks. The proposed method detected both the small and large test real leaks faster than the existing commercial LDS and its localization efficiency is quite comparable with the commercial software.

## 5.1 The Proposed Data Driven Leak Detection Algorithm

Figure 5.1 gives a summary of the proposed leak detection and localization algorithm. In this proposed algorithm, residual between the mass flow-rates of first and last nodes are continuously monitored. A positive residual implies a possible leak since according to the leak signature, mass flow-rate of the first node will be higher than the nominal flow-rate and mass flow-rate of the last node will be lower than the nominal flow-rate (Ostapkowicz, 2014). Positive residual in mass flow-rates can also be caused due to noise or malfunctioning of the flow sensors. Hence, the algorithm checks that the residual flow-rate is greater than a preset threshold for $T_d$ consecutive samples. The minimum of this threshold should usually be three times the standard deviation of noise of the measurements. The value of $T_d$ is selected to be between 5 to 10 in this case. If the sampling-time is high, then smaller

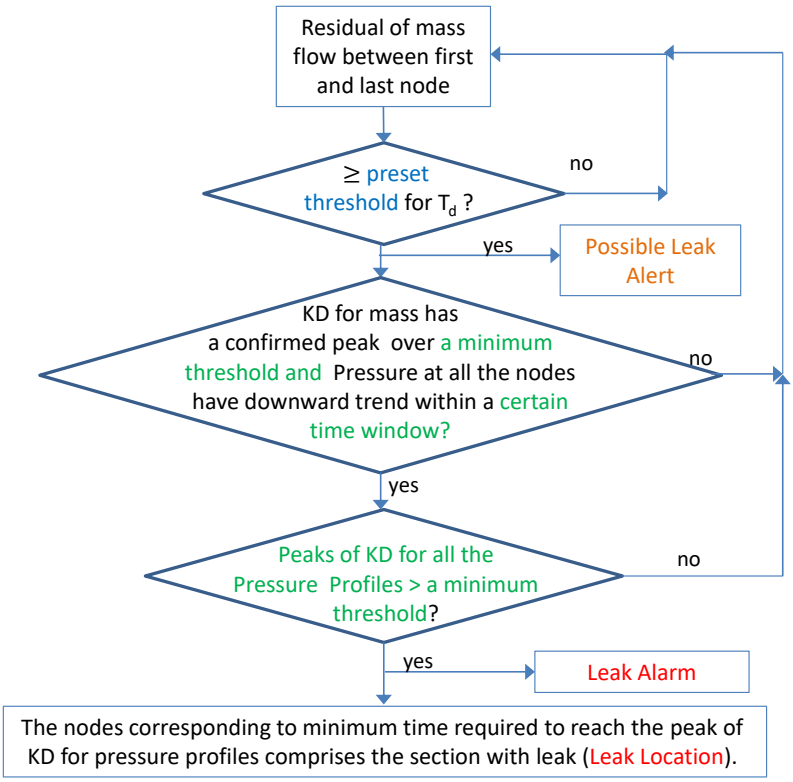


**Figure 5.1:** Framework of the proposed LDS algorithm. Here, $T_d$ is an 'on-delay timer' which means that when positive residual occurs, the algorithm will wait for $T_d$ samples before raising the leak alert.

$T_d$ is used. This step is necessary to reduce the number of false alarms due to outliers or noise. In case of flow sensor problems, the threshold of the residual terms should be set appropriately.

Once the residual is greater than the threshold for consecutive $T_d$ samples, a leak alert will be raised. Then the algorithm will start the KD calculation for the residual of mass flow-rate to confirm that a change has occurred. If the change is confirmed by detecting a peak in the corresponding KD score (the first peak condition), it will start checking the downward trends in all the available pressure measurements. This is the second leak signature (Ostapkowicz, 2014). If downward trends in all the pressure measurements are confirmed, the algorithm starts calculating KD for all the pressure measurements for a few samples around the time point where the alert was raised. The samples are chosen so that there are enough data points to calculate the KD. If peaks in the KD profile for all the pressure measurements are greater than a minimum threshold, then change is confirmed and leak alarm is raised (the second peak condition). These peak conditions are needed to reduce false alarms.

Once the leak alarm is raised, the next step of the LDS algorithm is to pinpoint the leak location. Once the KD is calculated for all the available pressure measurements, the time to reach the peak of KD for all of these individual pressure measurements are identified. The time corresponding to maximum value or peak of KD indicates when the true change has occurred in corresponding nodes. Since the nearest neighboring nodes (where pressure measurements are available) of the leak will have the highest impact of this leak (Geiger, 2006), the time to reach the maximum KD for these neighboring nodes will be the shortest. This is because, from the leak point, a negative pressure wave (NPW) will propagate in both the upstream and downstream directions with sonic (wave) velocity of the working fluid in the pipeline (Geiger, 2006) and this information is passed to the nearest upstream and downstream nodes the earliest. Thus, identifying the two neighboring nodes of the leak point, a smaller section of the pipeline is isolated where the possible leak could be.

Although the proposed algorithm cannot exactly identify the leak point, it can at least isolate a smaller section of the large pipeline which will reduce the effort of the plant personnel to look for the leak along the whole pipeline length.

## 5.2 Detection and Localization of Simulated Leak

In this case, the simulated pipeline was used as described in section 4.3.1. However, in this case, total simulation time was 1200 s. Also, this time it was assumed that two mass flow-rate measurements at node 1 ($W_{1,j}$) and node 21 ($W_{21,j}$) and five pressure measurements

available at nodes 1 ($p_{1,j}$), 6 ($p_{6,j}$), 12 ($p_{12,j}$), 18 ($p_{18,j}$) and 21 ($p_{21,j}$). This variation in the available measurement assumption was done to make sure that the proposed algorithms work well without being specific to the location of the sensors.

**Results from Simulation**

In this study, a leak was simulated along with transient operating conditions of the process. The Gaussian white noise was added to the measurements to make the simulation study close to a real scenario. A leak was introduced at the $900^{th}$ sample accompanied by a 5% step-up at $300^{th}$ sample and then a 5% step-down at $600^{th}$ sample. The step changes were carried out by changing the first boundary condition. The nominal flow-rate of the pipeline was 210 $kg/s$. For simplicity, the leak rate ($\dot{L}$) was kept constant at 10 $kg/s$ which is less than 5% of the nominal mass flow-rate. The leak was initiated at node 13 which is 24 km downstream from the first node.

Figure 5.2 shows the uniqueness of leak signature compared with the step-up and step-down. Panel *a* of this figure shows the mass flow-rates of node 1, 6, 12, 18 and 21 for this simulation. It is clear from this panel that when there is a step-up (as happened at $300^{th}$ sample) in the process, mass flow-rates of all nodes also increase. Similarly, during step-down at $600^{th}$ sample, flow-rates of all node have decreasing trends. But when the leak was initiated at $900^{th}$ sample, flow-rates of the upstream nodes of the leak node (node 13) are showing increasing trend and flow-rates of the downstream nodes have decreasing trend. The pressure leak signature is confirmed from panel *b* of this figure which shows lowering of pressures at all the nodes around the $900^{th}$ sample. It is to be noted that flow measurements for nodes 6, 12 and 18 are only available in the simulation environment and these measurements are not necessary for the proposed LDS.

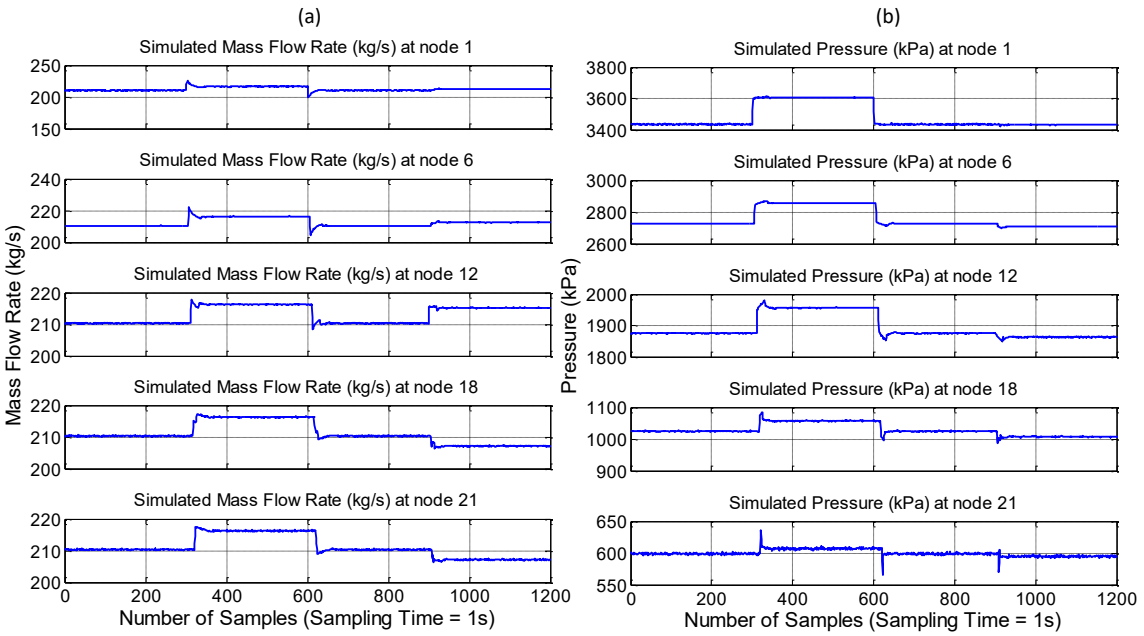


**Figure 5.2:** Panel (*a*) shows mass flow-rates at nodes 1, 6, 12, 18 and 21 with simulated leak at node 13 at $900^{th}$ sample accompanied by a step-up at $300^{th}$ sample and a step-down at $600^{th}$ sample. Panel (*b*) shows the pressure measurements at nodes 1, 6, 12, 18 and 21 for the same condition.

Figure 5.3 shows the mass flow-rates of first and last nodes, residual between them and the calculated KD of the residual mass flow-rate. One can see the persisting positive residual in mass flow-rates in the third panel of this figure from $908^{th}$ sample which is the first criteria to detect the leak by the proposed algorithm. The residual exceeded the threshold at $911^{th}$ sample and remained there for more than $T_d = 5$ consecutive samples. So, the algorithm raised leak alert for a possible leak at sample 916 ($t_{alert}$). The KD profile for this mass flow-rate discrepancy confirms the change with a peak generated at $923^{rd}$ sample and thus satisfies the first peak condition test.

For the above KD calculation and the following analysis to confirm that the alert is a true leak or not, the algorithm is provided with samples from $t_{alert} - 2(2m + k - 1)$ to $t_{alert} + 2m + k - 1$, so that the online KD algorithm gets sufficient data to cover the effect of transportation delay. The online algorithm waits for $2m + k - 1$ more samples before further analysis is done and this is the cause of detection delay. In this simulation, parameter $m$ was kept as 5 and $k$ was 2 for the KD algorithm. Before the KD calculation, either for mass or pressure, noise can be filtered out from the data applying a moving average filter of window size 3 to 5. If a larger window size is used, the detection delay will be larger, and if the window size smaller than 3 there will be less filtering benefit. Using the moving average filter depends upon the quality of data. If possible, use of a filter should be avoided

otherwise, some extra detection delay will be introduced.

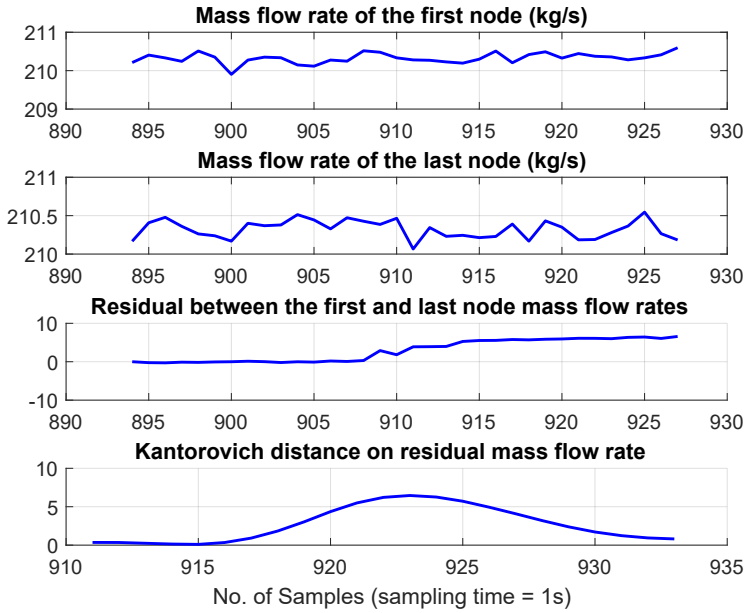


**Figure 5.3:** Mass flow-rates at the first and the last nodes, residual between them and KD profile of the residual for the simulated leak.

After issuing the leak alert, the algorithm starts checking the downward trend in all the available pressure measurements. Figure 5.4 shows that all the pressure measurements are showing downward trends after $900^{th}$ sample and nodes 12 and 18 are showing the downward trends earlier than any other nodes since these two are the nearest upstream and downstream nodes of the leak node (node 13), respectively. The leak is confirmed after satisfying the second peak condition test which refers to the formation of peaks in the KD profiles for all the available pressure measurements. The leak alarm is raised by the algorithm at 928 s which is less than half minute after the leak has been initiated.

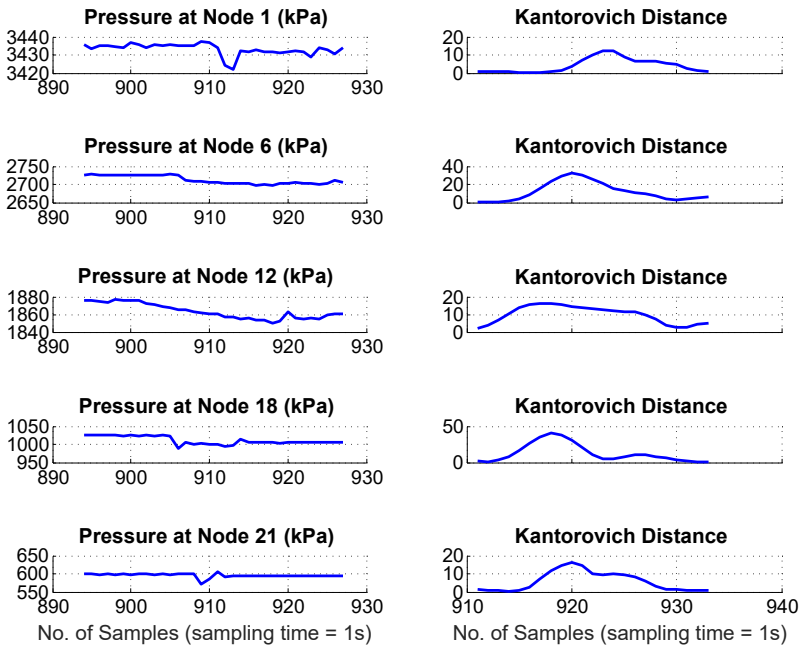


**Figure 5.4:** Downward trends in the pressure profiles to confirm the leak and their corresponding KD profiles for leak localization for the simulated leak at node 13.

The leak initiation time can be estimated by consulting the left panels of Figure 5.4, which shows that for node 12, the decreasing trend of the corresponding pressure measurement started at $902^{th}$ sample and this is the earliest indication of pressure drop among all the available pressure measurements. Therefore, the estimated leak initiation time is $902^{th}$ sample. The significance of this leak initiation time is that, when one also knows the time of shutdown the pipeline after detection of the leak, with the help of the knowledge of estimated leak rate, the leaked amount can also be estimated. This facility of estimating the leaked amount is not available in many commercial leak detection systems.

Leak localization is done by analyzing the times to reach the peaks of the KD profiles for all the available pressure measurements as seen in the right-panels of Figure 5.4. Figure 5.5 plots these change detection samples against the corresponding node positions. This figure indicates that peaks for KD profile were achieved faster in case of nodes 12 and 18 than the other nodes. So, it can be concluded that leak is within section 12-18 of the pipeline. Furthermore, one can see that node 12 has the lowest detection time and hence it gives the idea that node 12 is the closest node of the leak. This information narrows down leak location as the leak is in between section 12-18 and it is nearer to node 12 than node 18. Thus, this method can isolate a smaller section of the pipeline as the leak location.

The leak rate was estimated as 6.034 kg/s which is 2.87% of the nominal flow-rate. The leak rate is estimated by averaging the residual mass flow-rate between the first and last

nodes after the leak alert was issued. The algorithm could successfully differentiate the leak signature from two process transients namely the step-up and step-down. There was no false alarm in this case which is promising.

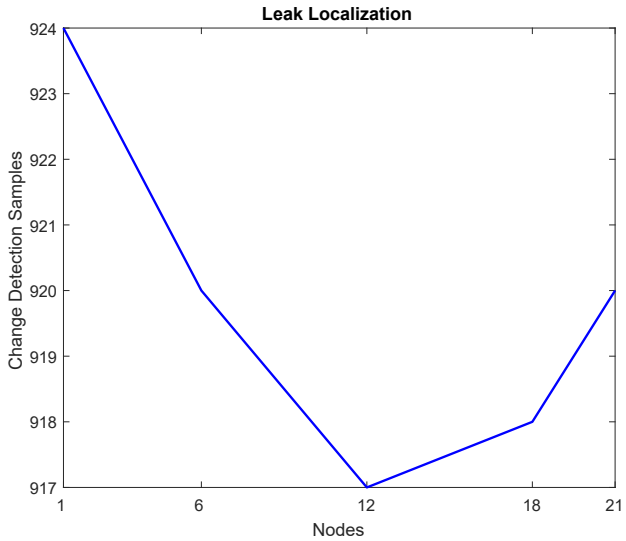


**Figure 5.5:** Localization of the leak using the minimum detection time concept for the simulated leak at node 13.

### 5.2.1 Real Time Detection of 1% Leak on the Industrial Pipeline

The leak size of the first test leak was about 1% of the nominal flow-rate. The sampling-time was 5 s. The proposed algorithm was connected with the real-time data via the OPC DA server through the Matlab OPC client. The detail of OPC connection mechanism is shown in Figure 5.6. This Figure shows that data from field RTUs or PLCs are coming to Modbus driver which is a part of the SCADA host. From Modbus driver, it goes to the OPC UA server and SCADA database. The OPC UA client then extracts the data for OPC DA server through which the Matlab OPC client gets the data. Matlab OPC client then feeds the data to the proposed LDS algorithm. The OPC UA client and DA server, Matlab OPC client and the proposed LDS algorithm are situated in a virtual machine which was the work-station in this case.

For this real-time test, the noise associated with the pressure and flow measurements were observed for few days and the following parameters were tuned:

- threshold for mass residual to raise the leak alert. Generally, this value is set as three times of the standard deviation of the noise associated with the mass flow measurements.

- threshold for the peak of KD of the mass residual. This threshold is required to ensure that the KD peaks in response to noise in mass flow measurement are not detected as leak. This threshold is crucial for the first peak condition.
- threshold for the peak of KD of the available pressure measurements. This is the threshold for the second peak condition. This threshold is associated with the noise in the downstream pressure measurement, since the downstream node has the lowest pressure in the system.
- an estimation of the transportation delay between the first and last nodes. This value can also be obtained from the experience of the operator.

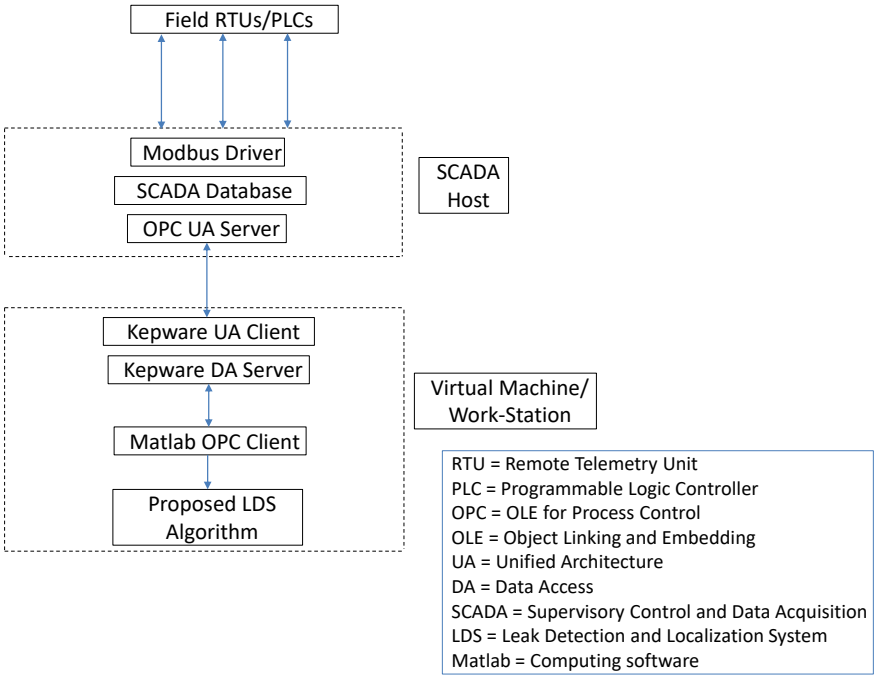


**Figure 5.6:** Mechanism of the real-time data connectivity with the proposed algorithm.

Two to three days operational data are sufficient to tune these parameters. These values need to be set carefully especially to detect a very small leak. Some trial and error may be required to tune and validate the selected parameters. The last panel of Figure 5.7 shows that the peak of the KD profile for this leak is very small if it is compared to the peak depicted by Figure 5.3.

The left bottom panel of Figure 5.8 shows that noise can affect the result when leak size is too small. The corresponding KD profile for this pressure measurement is shown in the

right bottom panel of this figure. It is clear that this KD profile has 3 peaks and the first two are due to noise. If the algorithm takes the maximum peak as usual for the localization purpose, it will give a false location because it will then take the first peak which concludes that the leak is around node 5 whereas the actual leak location is node 2.

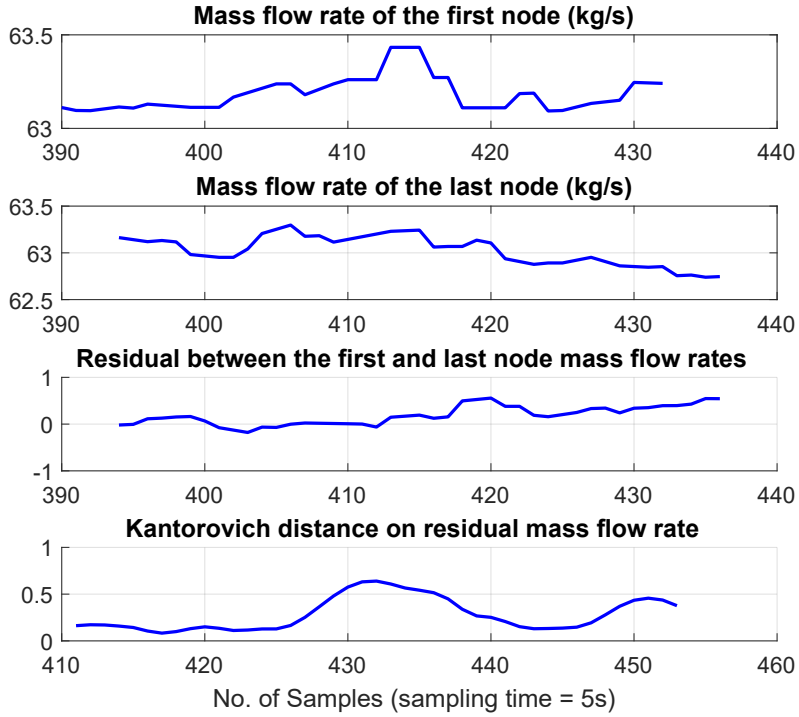


**Figure 5.7:** Mass flow-rates of the first and last nodes and their corresponding residual with calculated KD for the real-time test leak on the industrial pipeline with size 1% of the nominal flow-rate.

The proposed algorithm has the capability to overcome this issue. If the first peak of the KD profile of node 5 were due to leak, node 4 should have the second lowest time to reach the peak of the KD profile. But the reality was that node 4 needed the highest time among all the nodes to reach the peak in that case (Figure 5.8). Hence, node 5 cannot be the leak location. On the basis of this argument, the algorithm finally concluded that the leak location is between node 2 and 3 which shows the actual lowest times to reach the peaks. Figure 5.9 shows the leak localization for this small leak which was obtained by the proposed algorithm. This figure indicates that leak is in between node 2 and 3.

Table 5.1 shows the summary of this real-time leak test. One can see from the bracketed terms of the fifth row (Leak Rate) that the size of the actual leak is even lower than 1% of the nominal flow-rate. It is noteworthy that the time of detection and localization of the proposed method ($2^{nd}$ column) was much faster than the existing commercial LDS ($4^{th}$ column). The leak localization accuracy of the proposed algorithm is also quite satisfactory

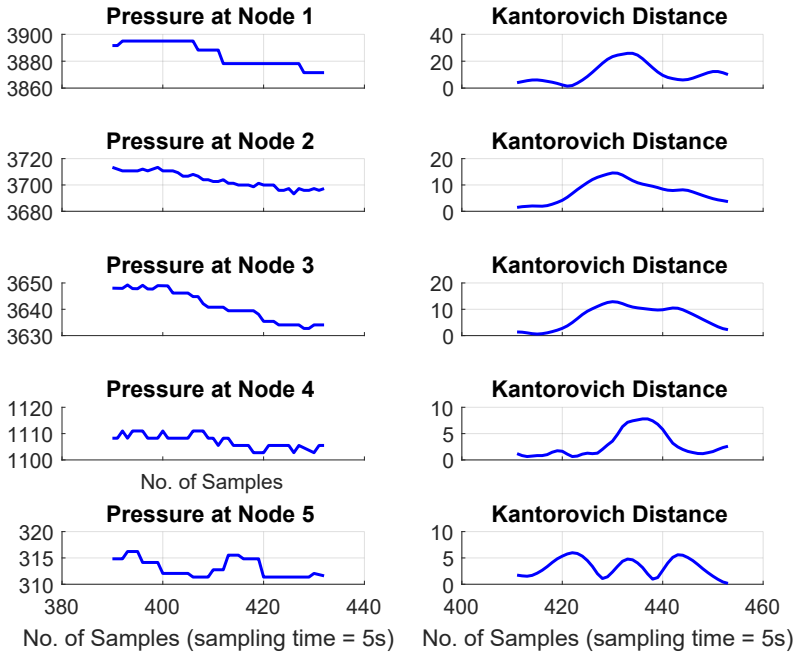


**Figure 5.8:** Available pressure measurements with calculated KD for 1% leak on the industrial pipeline in real-time.

if compared with true event ($3^{rd}$ column). The estimation of leak initiation time and leak rate of the proposed algorithm is quite comparable with the actual event.

| Event | Proposed Algorithm | Actual | Existing LDS |
|---|---|---|---|
| Leak Initiation Time | Sample 399 | Sample 398 | - |
| Leak Alert Raised | Sample 421 | - | - |
| Leak Alarm Raised | Sample 442 | - | more than 2000 sample |
| Leak Location | Section 2-3 | 3.626 km | 0.75 km |
| Leak Rate $\frac{m^3}{hr}$ | 2.95 (0.84%) | 3 (0.86%) | 2.82 (0.81%) |
| Leaked Volume $m^3$ | 8.85 | 9 | - |

**Table 5.1:** Comparison of the results of the proposed LDS algorithm with the existing commercial LDS along with the actual event for the 1% real-time leak test on the industrial pipeline.

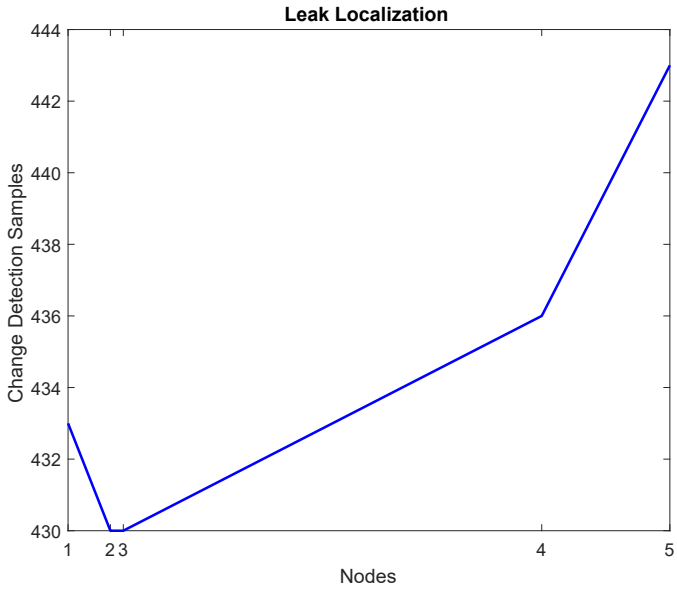


**Figure 5.9:** Leak localization for 1% leak in real-time. The nodes are presented according to their relative distances from one another. One can see that the leak location is between node 2 and 3 of the industrial pipeline.

### 5.2.2 Real Time Detection of 7% Leak on the Industrial Pipeline

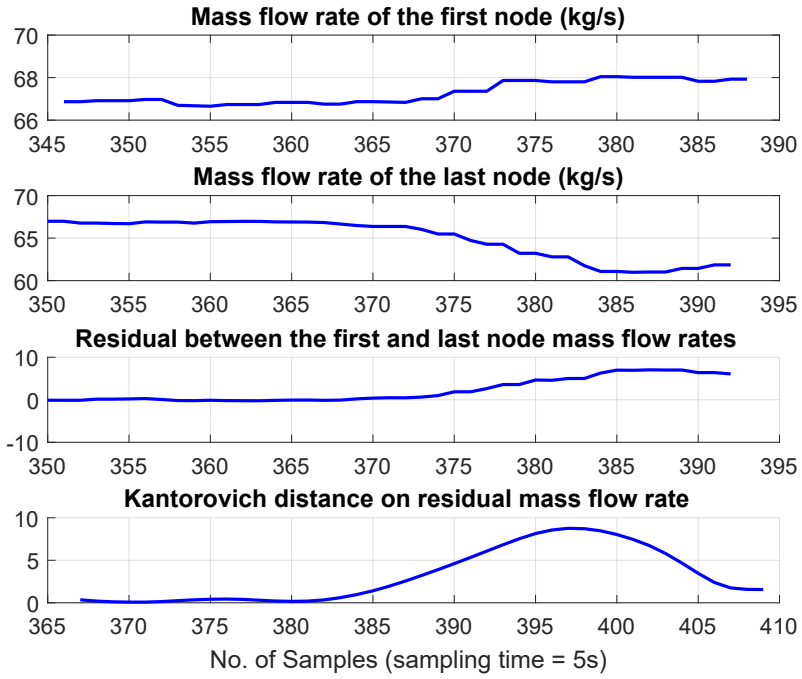


**Figure 5.10:** Mass flow-rates, their residual and the calculated KD of the residual for the 7% real-time test leak on the industrial pipeline.

The size of the second real-time test leak was about 7% to check the efficacy of the proposed algorithm to detect large leaks. Figures 5.10 to 5.12 demonstrates the efficacy of the proposed online data driven leak detection algorithm for this large leak. The updating time of the data from the OPC server was remained 5 s in this case. The first two panels of Figure 5.10 show mass flow-rates of the first and last nodes. From the third panel of this figure one can see that residual of mass flow-rates starts increasing from $373^{rd}$ sample and leak alert was raised at $378^{th}$ sample. The bottom panel shows the peak in KD for the residual mass flow-rate which confirms the change in mass flow-rate and satisfies the first peak condition.

The algorithm then starts looking for the decreasing trend of the available pressure measurements to confirm whether this is a true leak. The left-hand side panels of Figure 5.11 show the available pressure measurements in kPa units. The right-hand side panels of this figure show the corresponding calculated KD profiles. Peaks in these KD profiles confirm the change in the pressure measurements and thus the second peak condition is satisfied.

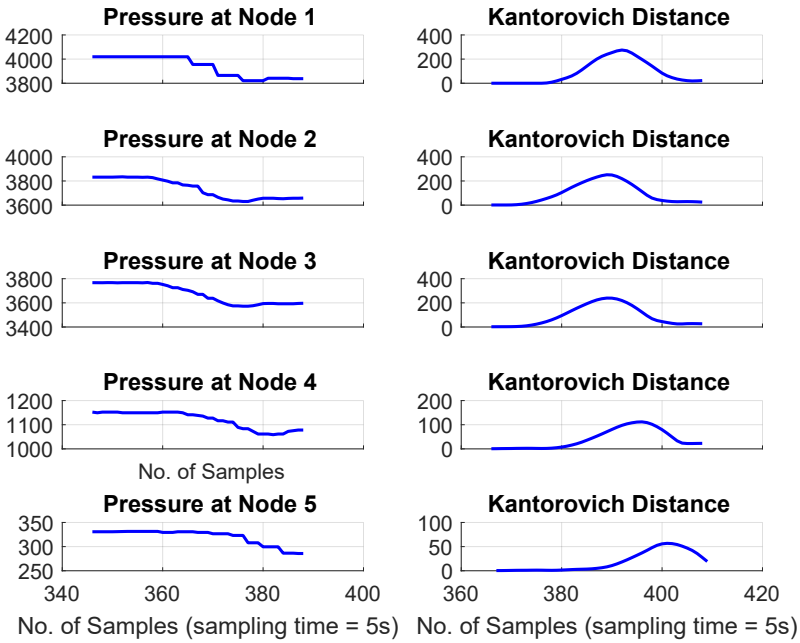


**Figure 5.11:** Available pressure measurements with the corresponding KD calculated for the 7% test leak on the industrial pipeline.

Figure 5.12 is obtained by plotting the change detection samples, which are times required to reach the peaks of KD for available pressure measurements, with respect the corresponding nodes. From this figure, it is clear that leak is around at node 2 since it has the minimum detection time. Since node 3 has the second smallest detection time, so it

can be concluded that the leak is in between nodes 2 and 3 and closure to node 2. The results from the proposed algorithm (column 2) are compared with actual event (column 3) and the existing commercial LDS (column 4) in Table 5.2. This table depicts that the performance of the proposed LDS is quite comparable with the existing commercial LDS. The leak alarm and localization times by the proposed algorithm are faster than those by the existing LDS. The proposed algorithm confirmed the leak almost 10 minutes earlier than the existing LDS and hence accuracy of the leak rate estimation for this varying leak rate case is also superior for the proposed algorithm. The estimated leak initiation time is also comparable with the actual event.

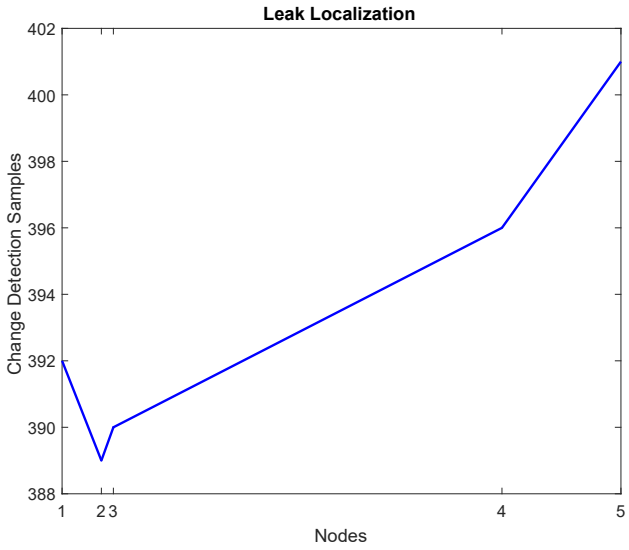


**Figure 5.12:** Leak localization using the minimum detection time concept for the 7% test leak in real-time.

| Event | Proposed Algorithm | Actual | Existing LDS |
|---|---|---|---|
| Leak Initiation Time | Sample 359 | Sample 364 | - |
| Leak Alert Raised | Sample 378 | - | - |
| Leak Alarm Raised | Sample 401 | - | Sample 526 |
| Leak Location | Section 2-3 (nearer to node 2) | node 2 (3.626 km) | 3.62 km |
| Leak Rate $\frac{m^3}{hr}$ | 27.23 (7.8%) | 25 (7.14%) | 5.01 (1.43%) |
| Leaked Volume $m^3$ | 5.69 | 5.425 | - |

**Table 5.2:** Comparison between the proposed algorithm and existing LDS along with the actual event for the 7% test leak on the industrial pipeline in real-time.

## 5.3 Concluding Remarks

A novel data-driven online pipeline leak detection and localization method has been proposed. The method is based on the Kantorovich distance. Efficacy of the proposed LDS has been demonstrated by successful detection and localization of a simulated leak in transient pipeline condition. Leak test results in the simulated environment also confirm that this algorithm can distinguish between a leak and a normal process transition. The proposed LDS scheme was also applied successfully to both small and large test leaks on an industrial pipeline in real-time. The algorithm was implemented in real-time via an OPC DA server. The times of leak alarm issuance and leak localization of the proposed algorithm are faster than the existing commercial leak detection system (LDS). Leak localization accuracy of the proposed method is also superior for the small leaks and for the large leak this accuracy is comparable with the existing commercial LDS. The number of false alarms was also inside the acceptable limit during these tests.

# Chapter 6

# Control Valve Stiction Detection and Compensation

Control valves are the hearts of any pipeline and process industries. Stiction in these control valves may cause a lot of off-spec product(s), waste of raw material and energy. In addition, due to repeated movement, the valve stem can wear out; such actuating equipment is expensive and just as expensive to install. This chapter discusses the stiction behavior following a formal definition. A novel data-driven stiction detection scheme has been proposed. The proposed method is based on the Kantorovich distance concept. Since stiction is a physical phenomenon the only way to get rid of its detrimental effect is either replacing the valve or change the valve packing during the next overhauling which may come in two-three years. An intermediate solution to this problem is the stiction compensation. A novel stiction compensation method has also been discussed in this chapter.

## 6.1 Formal Definition of Control Valve Stiction

The word '*stiction*' comes by combining two words '*st*atic' and 'fr*iction*'. Generally, it refers to the static friction between the valve stem and packing which hinders the valve stem from moving until a definite amount (which is generally referred as %-stiction) of controller output (OP) is accumulated. This accumulated controller output causes the actuator to overcome the effect of static friction and due to the integral effect of the controller output, a sudden jump (slip-jump) occurs in the valve positioner (Choudhury et al., 2008b).

Among all the definitions of stiction, probably the most practical and complete definition was given by (Choudhury et al., 2008b, 2005) as follows - *'The presence of stiction impairs proper valve movement, i.e., the valve stem may not move in response to the output signal from the controller or the valve positioner. The smooth movement of the valve in response to a varying input from the controller or the valve positioner is preceded by a*

*stick-band and an abrupt jump termed as slip-jump. Its origin in a mechanical system is static friction, which exceeds the dynamic friction during smooth movement of the valve.'*

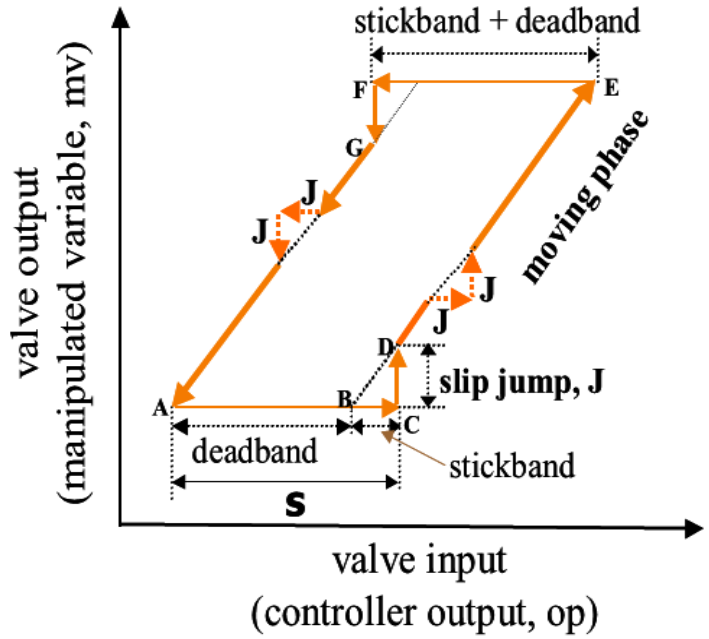


**Figure 6.1:** Typical input-output behaviour of a sticky valve Choudhury et al. (2008a, 2005)

Figure 6.1 represents the typical input/output behavior of a sticky valve. This phase plot can be divided into four phases - dead-band, stick-band, slip-jump and the moving phase. When the valve comes to rest or changes direction at point A, the valve sticks and it remains sticky until the controller output overcomes the dead-band (AB) and the stick-band (BC) of the valve. Then the valve jumps to a new position (point D) and continues to move. Due to very low or zero velocity, the valve may stick again between points D and E while traveling in the same direction. The section CD in the figure represents the slip-jump.

## 6.2 Mechanism of Valve Stiction

A feedback control loop (Figure 6.2) tries to regulate and maintain the process at the desired condition by comparing the actual process response (process-variable, PV) with the desired process response (setpoint, SP). If there is a deviation between these two signals, then the controller calculates the necessary corrective action and sends that signal to the control valve to open or close to nullify the error.

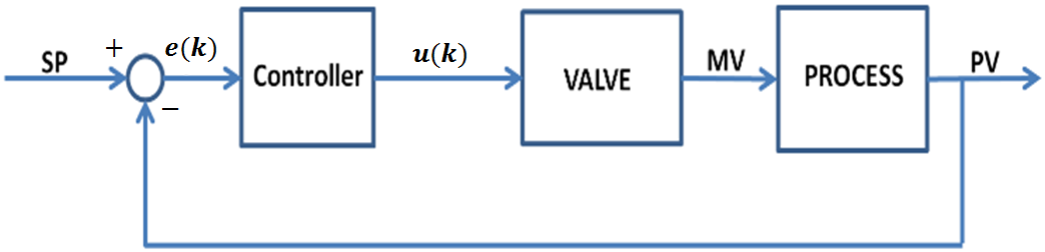


**Figure 6.2:** Schematic diagram of a simple feedback control strategy.

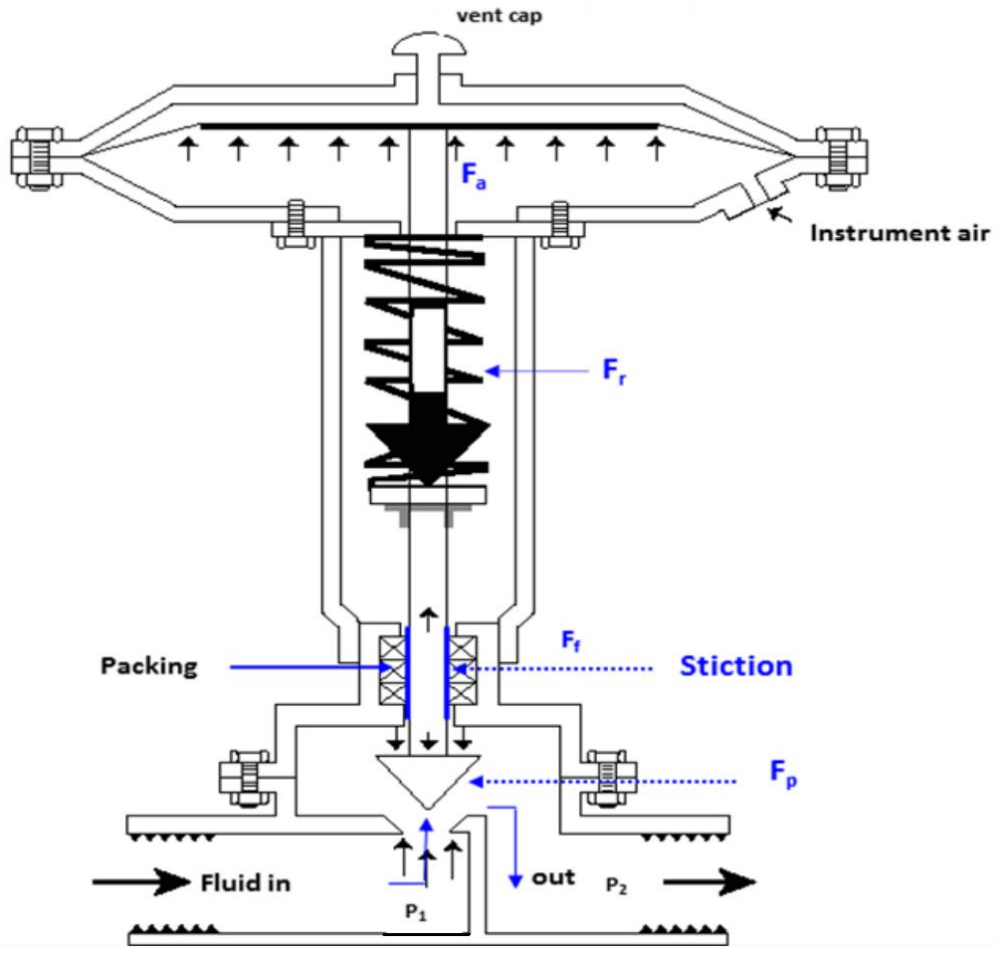


**Figure 6.3:** A detailed outline of a control valve.

However, if the control valve, typically the only moving mechanical part in a control loop, suffers from non-linearities especially stiction, the appropriate corrective action to mitigate the controller error cannot be implemented and may cause the loop to oscillate. A larger than normal static friction between the valve stem and valve packing (Figure 6.3) causes stiction. Generally, valve packing is used to prevent the fluid leaking from the

valve. If the packing is too loose, then fluid can leak; however if the packing is too tight, then excessive friction can prevent the valve stem from moving easily, that is not being able to respond to the controller signals. Accumulated OP (due to integral action) will finally generate enough force in the valve diaphragm to overcome the friction (this is defined as the parameter S = stick-band + dead-band). When this happens, the valve stem jumps causing the valve to move (defined as parameter J). Once the valve is in motion it may respond well to controller output (OP). However, when the valve changes direction such that at some point it may need to be stationary the stiction cycle repeats (see Figure6.1). Once again, the stick-band and the jump phenomenon occurs but in the opposite direction. The result of these repeated steps is the sustained oscillation in the process.

## 6.3 Detection of Stiction using the Kantorovich Distance Concept

Detection and quantification of stiction is a challenging job and a lot of work has already been done in modeling, detecting and quantifying stiction (Choudhury et al., 2005, 2008a; Xie, 2013).

In this study a novel method for control valve stiction detection has been proposed. The Kantorovich distance based change detection algorithm has been used to detect stiction. A first order plus time delay (FOPTD) process was used for this simulated case. The transfer function of the process is given as the equation 6.1.

$$G(s) = \frac{2e^{-5s}}{10s + 1} \tag{6.1}$$

A PID controller was tuned for the above process using the IMC tuning rules with the closed loop time-constant ($\tau_c$) as 5 s. In the first 1000 second of simulation, no stiction was there and PID controller was used to control the loop. At 1001 s, stiction was induced in the process using the two-parameter stiction model of Choudhury et al. (2005) with stiction parameter $S = 6$ and $J = 3$. A sustained limit cycle appeared in the process-variable (PV) as soon as the stiction was introduced. At time 2001 s, the stiction parameters were further changed as $S = 8$ and $J = 4$ to investigate that whether the Kantorovich distance algorithm can detect the variation in stiction. Since stiction is a non-linearity, the amplitude and frequency of limit cycle varies along with the change of operating zone of the valve. The top panel of Figure 6.4, shows these changes in the process-variable due to variability in stiction parameters.

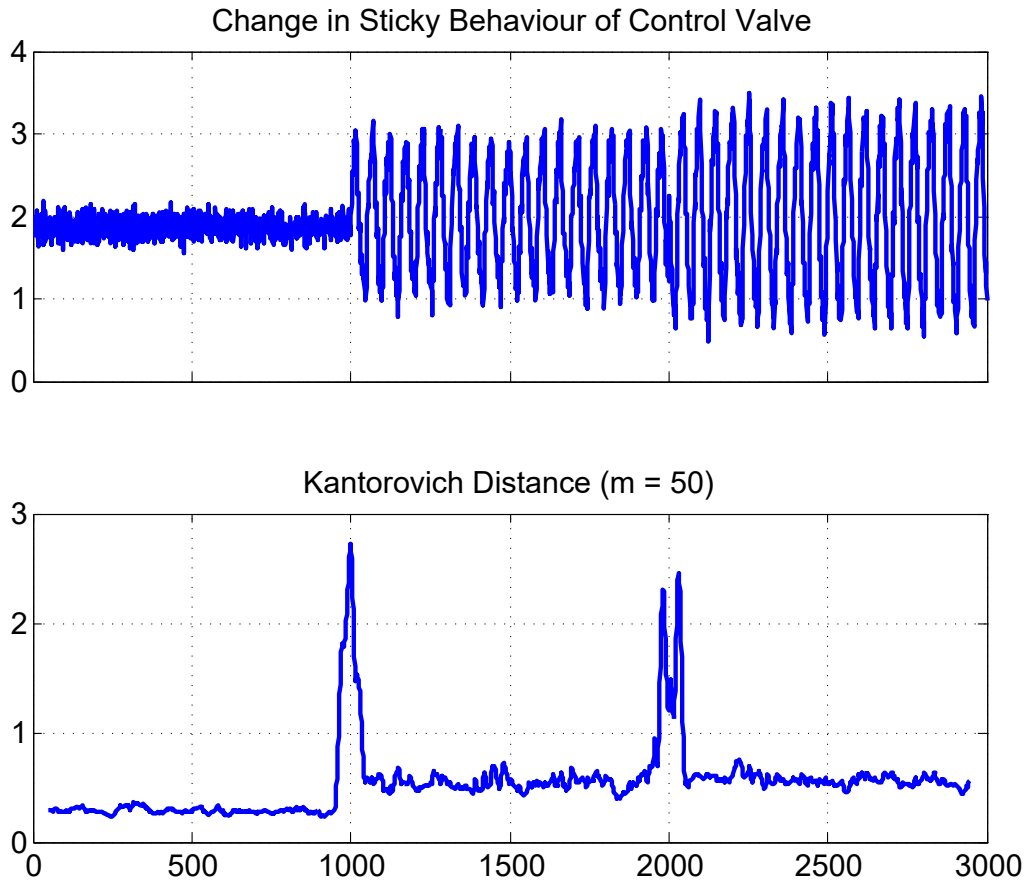


**Figure 6.4:** Detection of stiction using the Kantorovich distance concept.

The bottom panel of Figure 6.4, shows that the Kantorovich distance based change detection algorithm was successful in detecting stiction that was introduced at 1001 s. Once the stiction was detected, the KD score became low until the amplitude of limit cycle increased at 2001 s. Hence, the proposed stiction detection method is not only capable of detecting the stiction but also capable of detecting the change in stiction parameters which is an important feature for detecting stiction.

## 6.4 Compensation of Pneumatic Control Valve Stiction

Since stiction is a physical problem of the valve, after detecting stiction the only solution is to either repair (if possible) or replacement. But this is possible only during the plant shutdowns which generally occur once in every 2-3 years [ (Srinivasan and Rengaswamy, 2005)]. Therefore, compensation of stiction-caused oscillations can play a vital role to reduce the detrimental effect of stiction before the next shutdown occurs. Since more than 90% of the industrial control valves are pneumatic (Hägglund, 2002) and pneumatic valves

exhibit slower dynamics than servo-systems, compensation techniques such as dither or impulsive control (Armstrong-Hëlouvry et al., 1994) cannot be directly applied to process control loops (Srinivasan and Rengaswamy, 2005).

Though the available control valve compensation schemes are categorized into six categories in Brasio et al. (2014), they can be classified into two broad classes based on the use of stiction and valve models.

### Model Based Methods

The model-based schemes use -

- observer based method (Kayihan and Doyle, 2000),
- MPC based methods (Durand and Christofides, 2016b,a),
- optimization based methods (Srinivasan and Rengaswamy, 2006, 2008).

### Model Free Methods

In the model-free class, there are methods such as -

- knocker (Hägglund, 2002) and its variations (Srinivasan and Rengaswamy, 2005, 2006),
- constant reinforcement (Xiang Ivan and Lakshminarayanan, 2009),
- two-moves method (Srinivasan and Rengaswamy, 2008),
- controller tuning based methods (Gerry and Ruel, 2001; Mohammad and Huang, 2012; Mishra et al., 2014, 2015) and
- varying compensating signal methods (Cuadros et al., 2012; Arifin et al., 2014; Munaro et al., 2016).

The model-based methods require knowledge of the valve and often the process models which is a practical limitation for real-time application. Specially obtaining the valve velocity is not feasible in most real cases. Also, performing the calculation based on the models such as solving the optimization problems is time-consuming which is a roadblock to apply these methods in real-time. Therefore, a natural choice to overcome these drawbacks is the use of model-free approaches for which there are a wide variety of methods.

### 6.4.1 Model Free Stiction Compensation

The term model-free is used here to indicate that no friction model is required to generate the compensating signal. Some information about the control loop with the sticky valve is always required, e.g., the amount of stiction to be surmounted. Among the so-called model-free methods of compensation, one of the most common methods is the knocker method (Hägglund, 2002). In this method, short pulses of equal amplitude and duration in the direction of the rate of change of the control signal are added to the control signal (Figure 6.5). The pulses are characterized by three parameters and they cause the valve to move at all times, which unfortunately can wear the valve out well before its designated lifetime. Various modifications of this method have been described in (Srinivasan and Rengaswamy, 2005), (Srinivasan and Rengaswamy, 2006). The general equation for the compensating signal $u_c(k)$ in these methods is

$$u_c(k) = sign(u(k) - u(k-d))p(k) \tag{6.2}$$

where

- $u(k)$ is the controller output at the $k^{th}$ instant.
- $d$ is the delay to calculate the direction of $u(k)$. The delay $d$ is normally 1, but greater values (3 to 5) are used in general to reduce the effect of the noise.
- $p(k)$ is the pulse to be applied. The pulses can be constant, knocker-like, square wave or others.
- the $sign(.)$ function determines the 'sign' of the operands inside the parentheses.

The constant reinforcement (CR) (Xiang Ivan and Lakshminarayanan, 2009) method is similar to backlash compensation (Cuadros et al., 2012) and is a variation of the knocker method that uses only one parameter. In both cases the compensating signal is the varying controller signal with its sign multiplied by a constant, which keeps the valve moving all the time even more aggressively. In the two-moves method (Srinivasan and Rengaswamy, 2008), the $1^{st}$ move of the valve overcomes the stiction and second move brings the stem to its steady state position to reach the setpoint. The problem with this method is that it requires the knowledge of the compensating signal that yields the desired steady state position of the valve stem, which changes for different valve positions due to its inherent non-linearity. In this method, the move to reach the desired valve position that ceases oscillation is done in open loop. Also, when the setpoint changes, the procedure should be repeated. This is a practical limitation for industrial application.

The easiest way of model-free compensation is tuning the PI controller, and some tuning guidelines were outlined in (Gerry and Ruel, 2001) based on practical knowledge. P or PI controller tuning method for first-order self-regulating or integrating and FOPTD processes were investigated in (Mohammad and Huang, 2012) based on the the graphical analysis using the Nyquist plot and describing function analysis. However, in most of the cases, these tuning methods can reduce the oscillation but cannot remove it completely. Sometimes the amplitude of stiction is decreased at the cost of increase in the frequency of limit cycle. Moreover, quite often these methods introduce large overshoots and steady-state errors. However, all of the variations of the two-moves or PI tuning methods suggest that either ceasing or detuning the integral action of the PI controller might be helpful in combating stiction.

Based on the observations made on the tuning guidelines (Gerry and Ruel, 2001; Mohammad and Huang, 2012), methods have been attempted to reduce integral action automatically when the error between the setpoint (SP) and process-variable (PV) is near zero and increase the controller output to overcome the stiction when the valve is stuck. In (Mishra et al., 2014) a variable gain PI controller is used where the value of integral gain varies according to fuzzy logic on the basis of the change of error and the rates of change of error. The fuzzy logic increases the integral action when the error is high due to stiction which makes the controller output larger than normal to overcome stiction. In this sense, this method also lies in the same category as shown in Figure 6.5. In (Mishra et al., 2015) a non-linear function reduces the integral action to zero for very small errors and near zero rate of change of error. Two parameters are set by careful examination of process variability in noisy condition after performing open-loop experiments to estimate the parameters of the algorithm. The PI controller is tuned using a Differential Evolution algorithm and ITAE as cost function using data from an open loop step response. Thus, each setpoint requires specific tuning. This method seems to be somewhat complex to apply to real systems and also results for different setpoints in the same test are not shown.

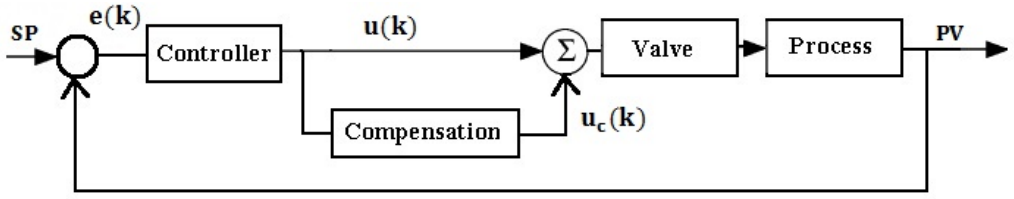


**Figure 6.5:** General schematic diagram for model-free stiction compensation methods.

A similar idea of stopping the integral action when the error is very low, is discussed

in (Cuadros et al., 2012). The pulses applied to compensate stiction were disabled when the derivative of the error was smaller than a threshold, and the integral action was also disabled so that small errors would not bring the oscillations back. However, the last pulse applied before these actions could move the valve away and violate the threshold, resuming the pulses, thus reducing the reliability of this method.

In this chapter, a novel model-free stiction compensation method has been proposed. The procedure of turning pulses on and off was replaced by application of pulses with amplitude proportional to the absolute value of the error. Thus, when the error ($e(k)$) between process-variable (PV) and setpoint (SP) is high, the compensating signal ($u_c(k)$) is also high which is added with the controller signal ($u(k)$) to get the valve move from its sticky position. When the error becomes small, the compensating signal also becomes small and hence is not able to move the valve stem. Therefore, the effect of stiction does not come back for small errors or noise. The next section describes the proposed method in detail. A comparison between the existing CR method and the proposed method is shown via an experimental run on laboratory setup equipped with a real sticky valve in the 'Computer Process Control (CPC)' laboratory of University of Alberta.

## 6.5 The Proposed Stiction Compensation Method

It is clear that both the knocker and the CR methods can reduce the detrimental effects of stiction, without any prior knowledge of the process and valve model, i.e., with minimal information. Also, they both can track the setpoint and reject disturbances. The problem with them is that in both cases the valve has to move a lot. In the case of CR, the valve moves more aggressively than the knocker method because of the higher amplitude of the pulses. On the other hand, characterizing the knocker signal requires three parameters to be chosen.

Hence, there is a clear need for a simple stiction compensation method that will not only reduce oscillations due to stiction but will also avoid unnecessary valve movement, will track the setpoint and reject disturbances with minimal information about the process and which can be applied online. In the present study, such a method has been developed which can be applied online to process plants.

The basic idea behind the proposed compensation scheme is that if the compensating signal is a function of error, then initially when the error is high due to the limit cycle, a higher amount of compensating signal will be added to the controller output to help the valve stem get rid of the sticky position. When the error reduces due to the action of the compensating signal, the amplitude of compensating signal will also reduce. So, at this

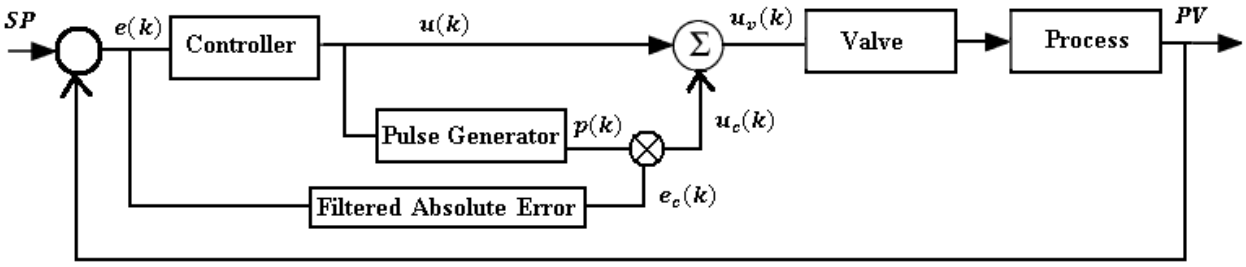


**Figure 6.6:** Stiction compensation strategies.

time the compensating signal will not be able to move the valve stem. Thus, this method not only breaks the limit cycle but also reduces the valve movement. The proposed scheme is shown in Figure 6.6.

The signal $p(k)$ is calculated using the knocker or CR methods. The amplitude of the pulses is $a = \frac{S}{2}$ in order to overcome stiction. The signal $e_c(k)$ is the filtered absolute error multiplied by a constant $\gamma$, which is between 0 and 1. The compensating signal is the product of $e_c$ and $p(k)$,

$$u_c(k) = e_c(k)p(k) \tag{6.3}$$

At this point, some important considerations are presented to ensure that this simple method works. The design requires only the measurements of the error and controller signals ($e$ and $u$). From these signals, one obtains $A_{OP}$, the amplitude of $u$, $A_E$ and $w_o$, the amplitude and frequency of oscillation of the error signal, respectively. These measurements are made under the assumption that the signals are oscillating due to stiction. The filter applied to the absolute error has a band-width related to the frequency of oscillation of $u$. The filter consists of two first order filters to make its implementation simple on commercial DCS. The time-constant ($\tau_e$) of this filter was chosen $\tau_e \geq \frac{1}{w_o}$, with greater values for noisy signals, to ensure that the oscillations are not attenuated. The value of $\gamma$ is chosen in order to satisfy

$$A_E\gamma \geq A_{OP} \tag{6.4}$$

$A_E$ is calculated using $u$ and the attenuation provided by the designed filter using $w_o$. The error signal is measured during the limit cycles. This value of $\gamma$ assures that the pulses have an amplitude large enough to overcome stiction. Since disturbances and changes in the setpoint can produce larger variations in the error signal, a saturation is applied to $e_c$, so that its limits are within the interval $[0, 1]$. The filtering and calculations on the error signal $e$ are illustrated in Figure 6.7. $e_c$ is the error from which the compensation signal $u_c$ is calculated and added to the controller output $u$. The EWMA (Exponentially Weighted

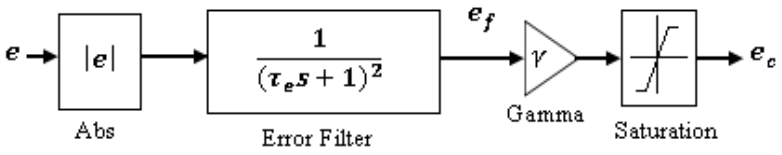


**Figure 6.7:** Signal flow path for computation of the error signal.

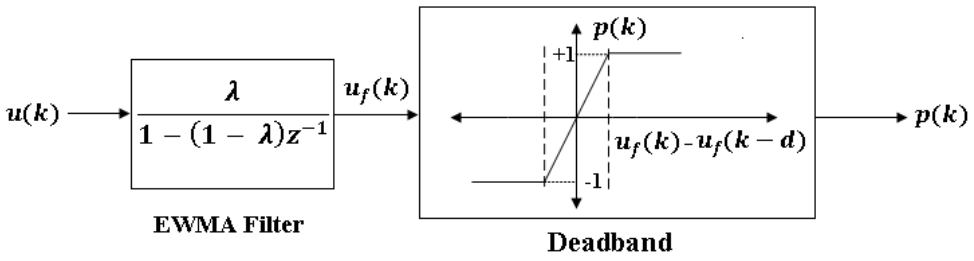


**Figure 6.8:** Signal flow path for computation of the signal $p(k)$.

Moving Average) filter applied to $u$ has the important objective of ensuring that the pulses are applied properly. Since this signal tends to be noisy, the direction calculated using its derivative can be affected, producing pulses in the wrong direction that increase the time for the compensation to eliminate the oscillation. An EWMA (Seborg et al., 2006) filter was used for $u$, since its implementation is quite simple on DCS and just one parameter is required for tuning according to the level of noise. An additional dead-band block is also applied before calculating $sign(\triangle u)$, to ensure that small variations on $u$ do not produce unnecessary pulses. Its value will be defined by $\delta_u$. The calculation steps to produce $p(k)$ from $u$ are illustrated in Figure 6.8.

Finally, a dead-band for PID control is also used. This strategy is usual in the industry when the integral action should be disabled for small errors. In this case, since a small error (offset) persists after compensation, causing the integral action to act and to bring the oscillation back, a small dead-band based on limits on the error amplitude is used. Generally, this is a built-in parameter in the PID blocks of almost all DCSs. This dead-band will be described by $\delta_{PID}$. The steps required for the design of the proposed stiction compensation scheme are described as follows:

1. Record $u$ and $e$ $(= SP - PV)$ during some periods of oscillation and obtain $A_{OP}$, $A_E$ and $w_o$.

2. Calculate $\gamma \geq \frac{A_{OP}}{A_E}$ (as per equation 6.4).

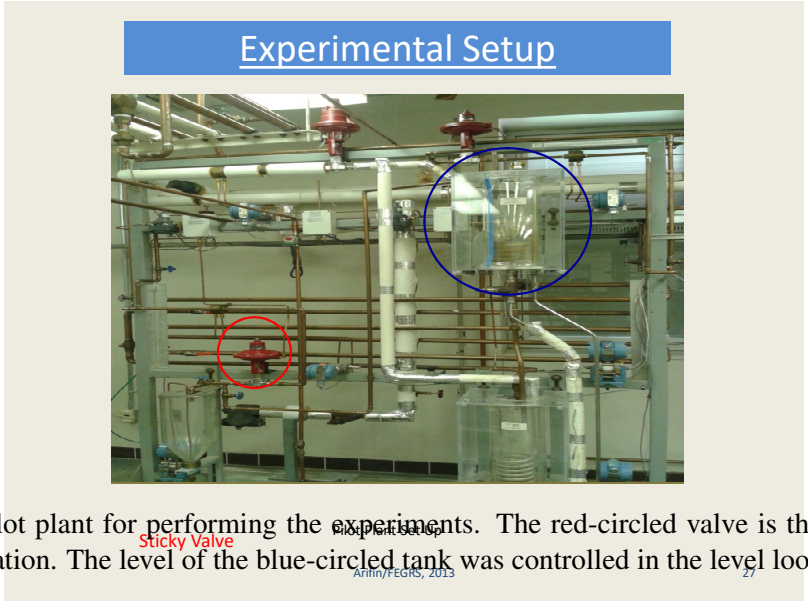


**Figure 6.9:** Pilot plant for performing the experiments. The red-circled valve is the sticky valve under consideration. The level of the blue-circled tank was controlled in the level loop.

3. Calculate time-constant for error filter, $\tau_e \geq \frac{1}{w_o}$ [unit = $\frac{sec}{rad}$].

4. Select $\lambda$ in the EWMA filter to reduce noise. Values around 0.5 are a good choice in general. It depends upon how noisy the controller output signal $u$ is.

5. Calculate dead-band for $p(k)$, $\delta_u \leq 0.1 A_{OP}$.

6. Calculate dead-band for PID controller, $\delta_{PID}$, based on maximum error. A reasonable choice is $\delta_{PID} > 0.2\ max(e)$ (maximum after applying compensation).

## 6.6 Experimental Validation

To validate the proposed stiction compensation method some experiments were performed on a level loop of a computer interfaced pilot plant in the computer process control laboratory at the University of Alberta (Figure 6.9). The pilot plant is equipped with Delta V DCS system.

### 6.6.1 Experiments with the Existing CR Method on a Level Loop

Figure 6.10 shows the experimental evaluation of the constant reinforcement (CR) scheme for compensation of stiction. From this figure, it is clear that the CR method could eliminate the high amplitude low-frequency oscillations at the expense of small amplitude high-frequency oscillations and aggressive valve movements. The knocker method also shows similar results, but it is less aggressive than the CR method regarding the valve movement.

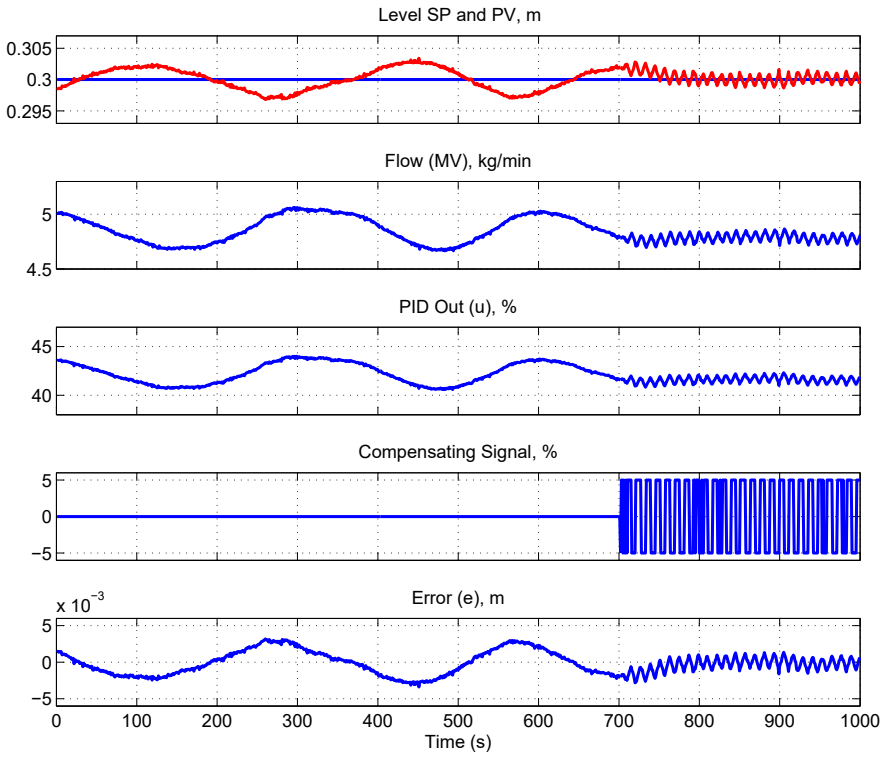


**Figure 6.10:** Experimental evaluation of the CR compensation scheme. The scheme reduces the amplitude of oscillation but increases limit cycle frequency.

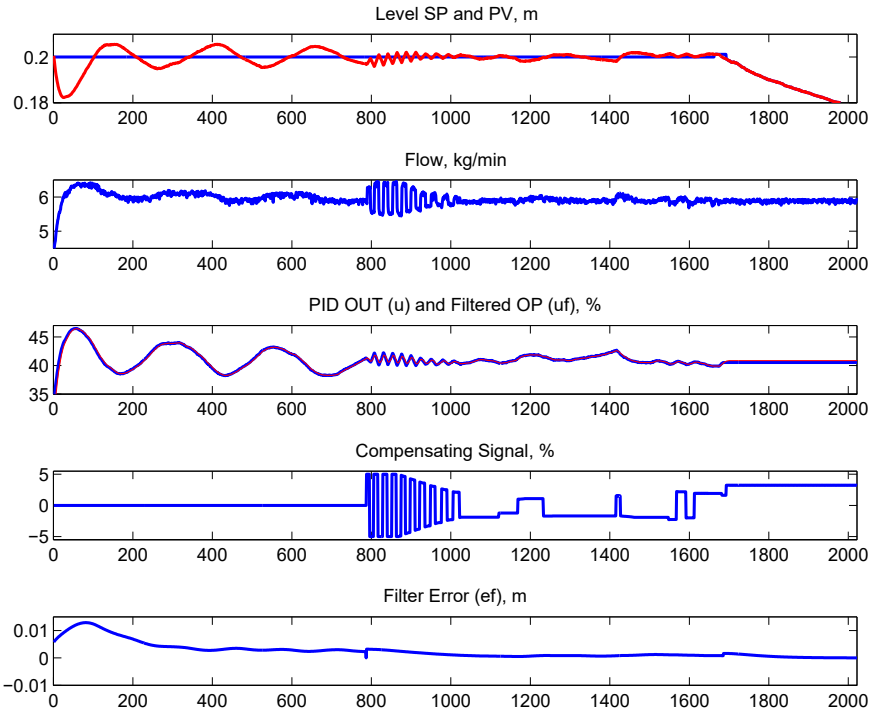


**Figure 6.11:** Experimental Result for the level loop. There was a sustained oscillation in the process up to 786 s. Then the proposed scheme was activated at 787 s. The proposed scheme was able to arrest the effect of stiction. At 1692 s the process was set to manual mode to show the presence of measurement noise.

### 6.6.2 Experiments with the Proposed Method on a Level Loop

The proposed compensation scheme was applied to the same level loop (as shown in Figure 6.9). In this case, the proportional gain was 2 and reset-time (integral action time-constant) was 50 s for the PI controller. The scan-rate of the level measurement in the Delta V system was selected to be 1 s.

The result is shown in Figure 6.11. Here, for the first 786 s no compensation scheme was active; there after the proposed compensation scheme was initiated. In this case $A_{OP} = 2.79$, $A_E = 0.003$ and $w_o = \frac{2\pi}{267}\frac{rad}{s}$. The parameters for the proposed scheme were calculated as $\tau_e = \frac{1}{w_o} = 42$, $\gamma = 930$, $\lambda = 0.6$, $\delta_{PID} = 0.001$ and $\delta_u = 0.25$. It is clear that the proposed scheme was able to break the oscillation due to stiction. To check the effect of the noise, the valve was set to manual mode at 1692 s and it is clear that there exist some measurement noise and that the valve was not moving.

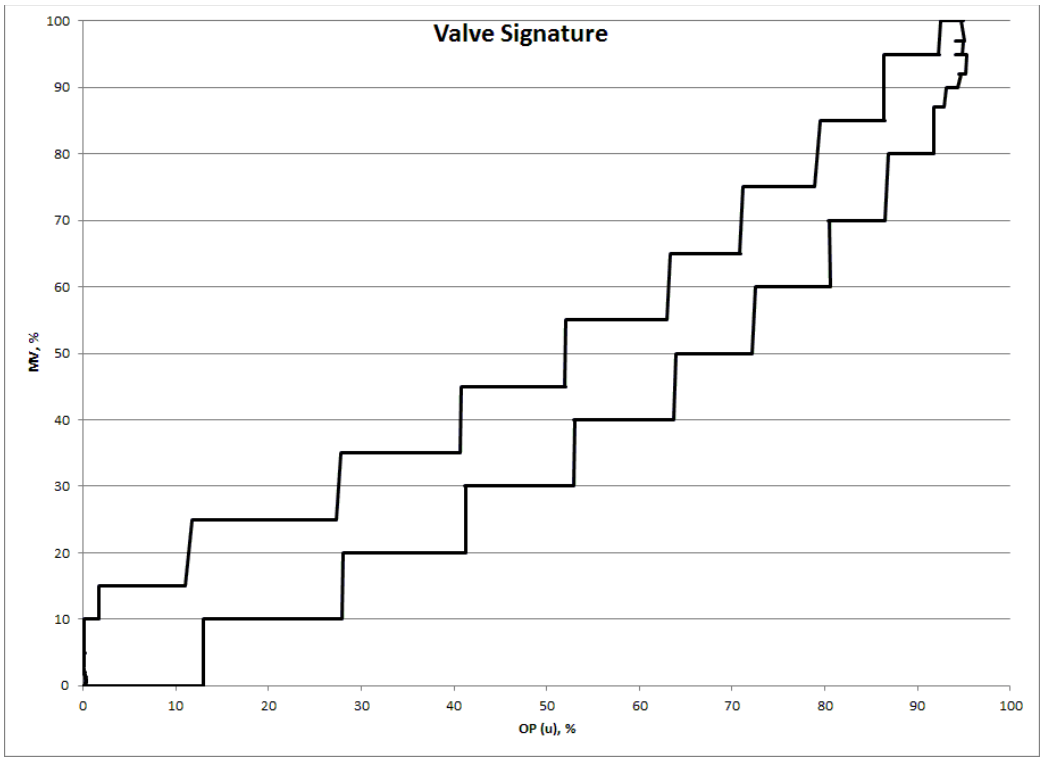


**Figure 6.12:** Signature of the sticky control valve (MV vs. OP plot).

### 6.6.3 Experiments with the Proposed Method on a Flow Loop

The scheme was applied to control the inlet flow-rate of the same tank shown in Figure 6.9 to show the applicability of the method for fast process dynamics. Since the flow loop is faster, in this case the Delta-V block scan rate was selected to be 0.1 s. This was to make sure that there is enough time for calculation before the valve changes its direction and the compensating signal was applied at the right moment. In this case the proportional gain was 1 and reset time was 0.6 s for the PI controller. In the tests for the level loop the flow was around 5 $kg/min$, while in the tests for flow loop it was around 2 $kg/min$. The signature obtained for the valve showed that stiction changes considerably for different points of operation, changing the amplitude of the controller signal (Figure 6.12). Figures 6.13 and 6.14 represent some of the results from these experiments. In Figure 6.13, over the first 169 s, there was no compensation; from 170 s onwards the CR compensation scheme was applied. Using data from the first 169 s, the following values were obtained: $A_{OP} = 15.92$, $A_E = 0.68$ and $w_o = \frac{2\pi}{7}\frac{rad}{s}$. Using these values and the expressions in the six steps of the compensation scheme, one obtains: with $\tau_e = 1.1$ s, $\gamma = 22.9$, $\lambda = 0.6$, $\delta_{PID} = 0.1$ and $\delta_u = 0$. The value of $\tau_e$ was 5 to further reduce noise effects. From 513 s onwards, the proposed compensation scheme was active. From 689 s onwards, a setpoint change was

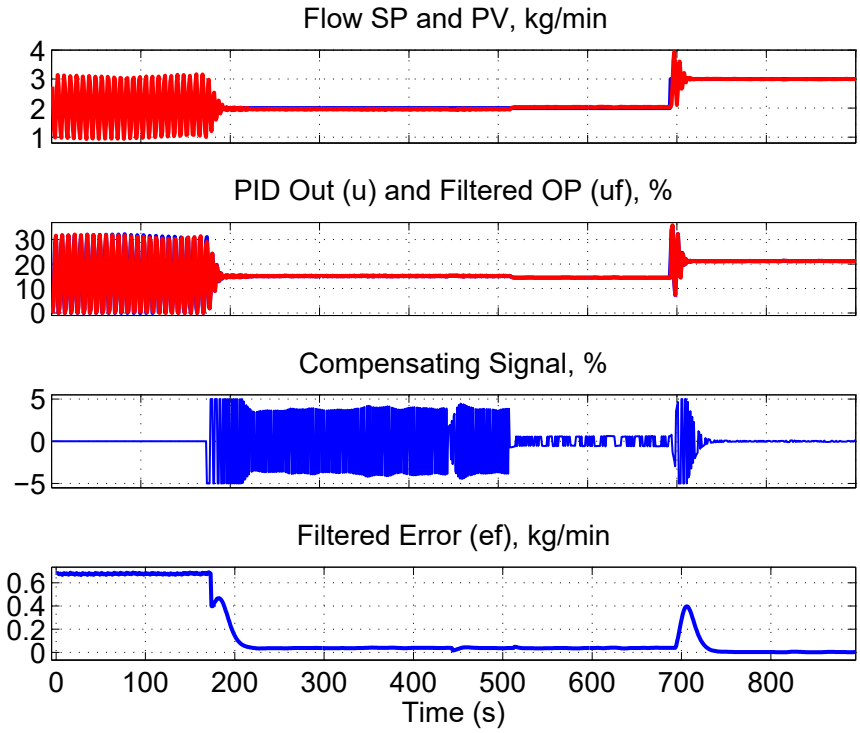


**Figure 6.13:** Effect of CR (170 to 512 s) and the proposed compensation (513 s onwards) on the flow loop. It shows that in the case of CR scheme the compensating signal is higher and hence the valve movement is also higher.

implemented with the proposed compensation scheme at work. The figure does show that the proposed scheme works well also during a step change. Figure 6.14 is the zoomed-in version (from 350 s to 650 s) of Figure 6.13 to clearly show the difference between CR and the proposed case. After the proposed scheme was activated, the valve movement reduced significantly.

Figure 6.15 shows the effect of setpoint change while the proposed scheme was at work. At 18 s the setpoint was changed from 2 $kg/min$ to 5 $kg/min$ with the proposed scheme being active. This figure shows that the proposed compensation scheme also works well when a new disturbance affects the system. It is to be noted that at 297 s, a disturbance affected the setpoint. The proposed scheme handled the disturbance quite effectively. From Figures 6.11, 6.13 and 6.15 it is clear that the filtered error provides a good compensation mechanism to change the amplitude of the pulses to handle stiction and to have the PV converge in the vicinity of the SP. Though initially, the valve moved a lot, as the PV neared the SP, the valve movement stopped and the valve there after only moved when a new error was introduced. Figure 6.16 is a zoomed-in version of Figure 6.15. It shows that

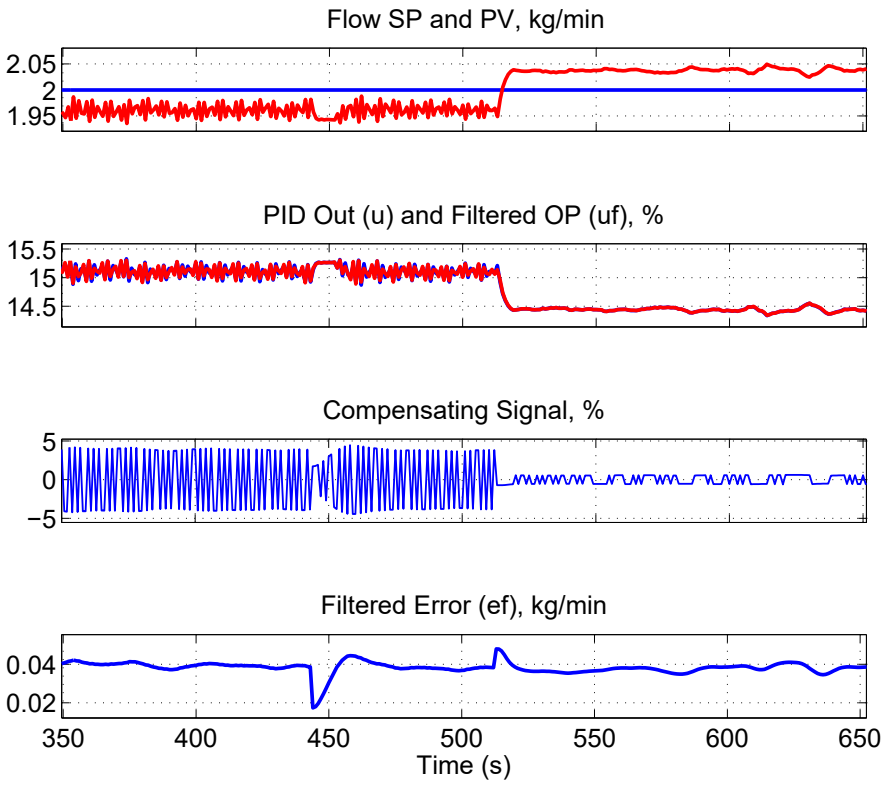


**Figure 6.14:** Zoomed-in Version of Figure 6.13. Notice that the valve movement is significantly reduced with the proposed scheme.

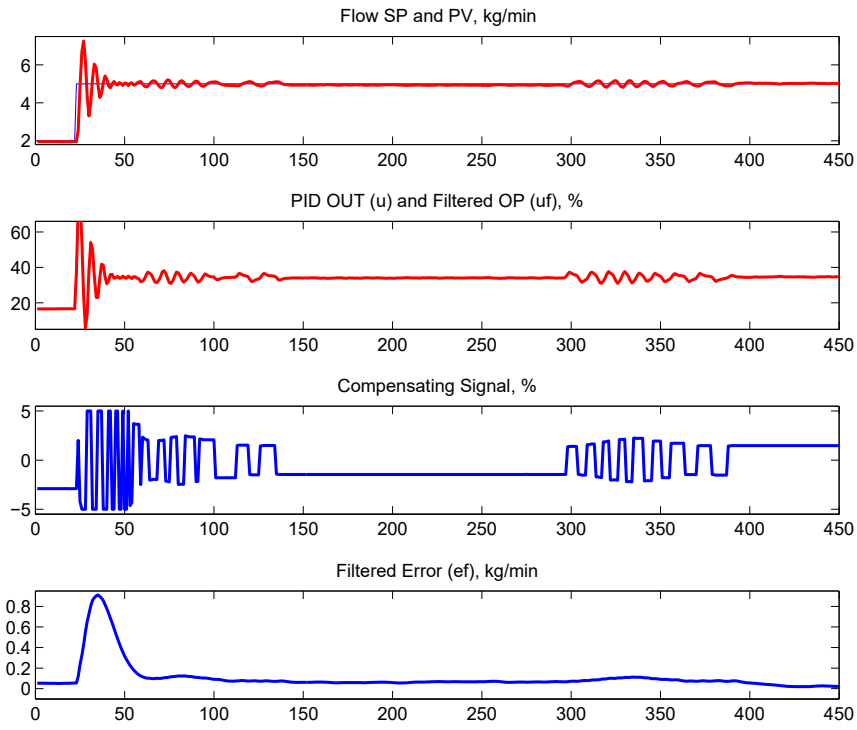


**Figure 6.15:** Setpoint tracking (18 s) and disturbance rejection (297 s) properties of the proposed compensation strategy. The compensating signal is added when there is an error. But when the error reduces, the compensating signal is automatically reduced.

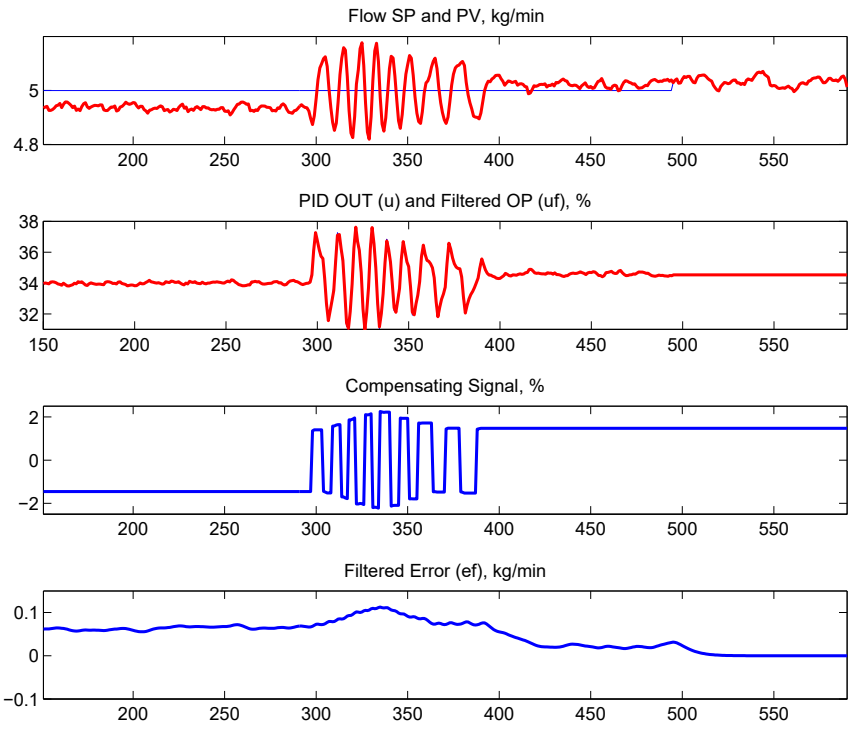


**Figure 6.16:** Zoomed-in version of Figure 6.15. At 297 s, a disturbance was introduced and the compensation scheme tackled this disturbance and took the process to the desired condition at 392 s. The process was set to manual mode at 493 s to show the presence of measurement noise.

the proposed scheme handled the disturbance quite effectively and at 392 s the effect of disturbance was minimized. At 493 s, the valve was set to manual mode to show that small variations on flow come from noisy measurements, not from valve movement.

## 6.7 Concluding Remarks

This chapter describes novel method for detecting and compensating valve stiction. The main contributions can be summarized as:

1. a novel data-driven control valve stiction detection method has been developed using the Kantorovich distance algorithm which can also detect the variability in stiction parameters.

2. a novel valve stiction compensation technique has been developed which can not only eliminate oscillations without aggressive control actions but also can achieve good setpoint tracking and disturbance rejection. It takes the compensation action based on the amount of error between setpoint (SP) and process-variable (PV), which is the novelty of this method.

3. the proposed compensation method neither requires a process model nor any input from the operators.

4. the proposed compensation method is applicable to both fast and slow control loops such as flow and level loops.

5. the proposed compensation method has been successfully implemented on a pilot-scale laboratory tank system which was interfaced using a commercial Delta-V DCS system.

# Chapter 7

# Stiction Compensation : Ramp Modulation of Compensating Signal

The performance of stiction compensation method described in Chapter 6 (Arifin et al., 2014) reduces drastically if the error limits are not achieved before the amplitude of pulses becomes higher than half of the friction. In this situation, the error keeps increasing proportionally to the amplitude of the pulses, and there is no convergence (Figure 7.1). To overcome this limitation an improved version of stiction compensation method was proposed in (Munaro et al., 2016). A unidirectional search in the amplitude of the pulses is performed and when the corresponding absolute error becomes smaller than a specified threshold, the pulses are ceased and the integral controller action is disabled. Whenever the error becomes greater than the limit, the integral action is enabled again. Several parameters were specified based on author's experience, but the method was not applied to a real sticky valve. Instead, a sticky valve was emulated using a micro-controller which was connected to an industrial controller, where the compensation algorithm was implemented in an industrial controller. The concept of ramp type modulation of the compensating signal that was used in this work has some similarity with the ramp-type modulation of pressure drop to control the automotive brake valve (Lv et al., 2017; Zhang et al., 2014). The difference is that the ramp type modulation of compensating signal was not only to reduce the compensating signal but also to find out the best value of the compensating signal to combat stiction with minimum valve travel and IAE. So, the ramp modulation used in (Munaro et al., 2016) works to find out a possible optimum value of the compensating signal.

In this chapter, an improved version of the method described in (Munaro et al., 2016) is proposed. Here, the selection of these parameters is reviewed and simplified. The flexibility allowed in these choices is analyzed via simulations. This time, the performance of the proposed method was also compared with other stiction compensation algorithm. The implementation of this scheme on the industrial DCS system using IEC 61131-3 standard

allow its evaluation on different industrial controllers. The application of the proposed method to a flow and a level loop of a pilot plant with equipment similar to those found in industry illustrates the practicality and utility of the proposed scheme in real industrial control loops. A real sticky-valve with built-in valve positioner is used, and it is shown that the intrinsic position control of the valve may degrade stiction compensation schemes that may lead to a small error in the process-variable.

Six criteria for a successful stiction compensation method to work in industrial control loops are presented in Table 7.1, based on a previous list from (Cuadros et al., 2012). From this table it is clear that the Knocker (Hägglund, 2002) and the CR (Xiang Ivan and Lakshminarayanan, 2009) methods have all the criteria of a stiction compensation method except reduction of wear. The two-moves method and its variations have fulfilled only two criteria. The varying integral gain methods (Mishra et al., 2014, 2015) fulfill only three criteria. The proposed varying compensating signal method fulfills all the criteria listed in Table 7.1.

**Table 7.1:** Comparison of available stiction compensation methods

| Criteria | Knocker | CR | Two-Move | Non-linear PI | Proposed |
|---|---|---|---|---|---|
| Reduction of oscillations | ✓ | ✓ | ✓ | ✓ | ✓ |
| Reduction of wear | × | × | ✓ | ✓ | ✓ |
| No open loop test required | ✓ | ✓ | × | × | ✓ |
| Tracking of setpoint changes | ✓ | ✓ | × | × | ✓ |
| Disturbance rejection | ✓ | ✓ | × | ✓ | ✓ |
| Implementation in industry | ✓ | ✓ | × | × | ✓ |

The remaining of this chapter is as follows - Section 2 describes the proposed method in detail. Section 3 describes how parameters are chosen for this algorithm, using simulation. In section 4, application of the method in a real plant is discussed along with application results. Finally, section 5 gives concluding remarks.

## 7.1 Proposed Variable Amplitude Stiction Compensation Method

The main idea behind the proposed method is to perform a unidirectional search for the amplitude of pulses that brings the error within specified limits. This approach is supported by Figure 7.1, using the simulation and parameters described in Section 7.2. This figure shows the effect of the amplitude of the pulses on the integral absolute error (IAE) and the

valve travel (VT) measured using equations 7.1 and 7.2 respectively.

$$IAE = \sum_{k=1}^{N} |e(k)| \tag{7.1}$$

$$VT = \sum_{k=1}^{N} |x(k) - x(k-1)| \tag{7.2}$$

where, $e(k)$ is error between setpoint (SP) and process-variable (PV) at the $k^{th}$ sample and $x(k)$ represents the valve position at $k^{th}$ sample. The window to calculate the IAE and VT goes from the time of start of the compensating signal till the end of simulation. $N$ indicates the total number of samples considered to calculate VT and IAE.

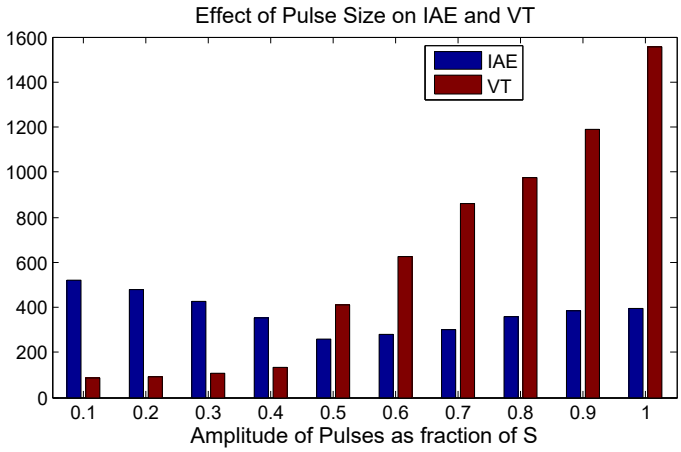


**Figure 7.1:** Effect of the amplitude of the pulses.

The value from the x-axis of Figure 7.1 is multiplied with $S$, that is an approximation of stiction, resulting in the amplitude of the pulses that are applied. One can see that IAE starts decreasing with the increase of pulse magnitude and reaches its minimum when the applied pulse is $0.5S$. After that, IAE increases with the increase of pulse magnitude. The valve travel increases when the amplitude of the applied pulse increases, and this increase is more pronounced when the amplitude is greater than $0.4S$. Since there is no optimum pulse amplitude when both IAE and VT are considered, the proposed algorithm searches the amplitude to reduce the error to a specified limit and then ceases the pulses. The proposed algorithm is represented by the flowchart shown in Figure 7.2, which is contained in the compensator block of Figure 7.3. This figure is very similar to Figure 6.5. The input $p(k)$ and output $\eta(k)$ of this block will be explained after presenting the flowchart.

When compensation becomes active, compensation pulses are computed using 7.3.

$$u_c(k) = sign(u(k) - u(k-d))p(k)r(k) \tag{7.3}$$

This equation is similar to 6.2, with the addition of $r(k)$, a ramp signal that starts with value 0.5, ends with value 0.2, whose slope is given by

$$\Delta r = \frac{0.3}{n_r} \tag{7.4}$$

$$n_r = floor(\frac{4\tau}{T_s}) \tag{7.5}$$

where $\tau$ is the approximate value of time-constant of the control loop and $T_s$ is the sampling interval of the loop. Thus, the ramp modulates the amplitude of the pulses, so that they have amplitude large enough to help overcome stiction and are not too large that can cause large error (IAE) and valve travel (VT). The simulation shown in Figure 7.1 supports this choice of limits for $r(k)$. The slope is chosen so that every cycle of the ramp lasts one settling time ($4\tau$), so that the control loop can be affected by the pulses with varying amplitude. Smaller values of $n_r$ reduce the chance for the pulses to be effective, while larger values increase the time for the compensation to work, reducing the performance indexes. A good variation for this choice of $n_r$ is allowed, and will be shown in the simulations of Section 7.2.

In the flowchart, at every sample time $k$, the amplitude of the pulses is reduced according to the new value of $r(k)$ and the signal $u_v(k)$ is applied to the valve. The signal $u_v(k)$ is the sum of PI signal $u(k)$ and compensating signal $u_c(k)$ given by equation 7.3. If the absolute value of the error becomes smaller than the specified limit $\delta$, this event is counted, and after $n_e$ counts the pulses are ceased. To cease the pulses, the amplitude of the ramp $r(k)$ is set to zero. Also, the dead-band for the integral action of the PI controller, represented by signal $\eta(k)$ in Figures 7.2 and 7.3, is set to $\delta$. This action is required to prevent the integral action of the controller to bring oscillation back, and is very common in stiction compensating schemes, as discussed in Section 1. If the error becomes greater than the threshold, the pulses are resumed making $r(k) = 0.5$ and setting the dead-band of integral action of PI controller to zero, i.e., $\eta(k) = 0$. A suggested value for $n_e$ is given by

$$n_e = floor(\frac{\tau}{T_s}) \tag{7.6}$$

This parameter is chosen to assure that if the error is within the specified limits during $n_e T_s$ seconds, it will remain there if the pulses are ceased. Therefore, this parameter is related to the dynamics of the control loop, and one time-constant is a good choice. Simulations will be performed to show the flexibility in choosing this parameter.

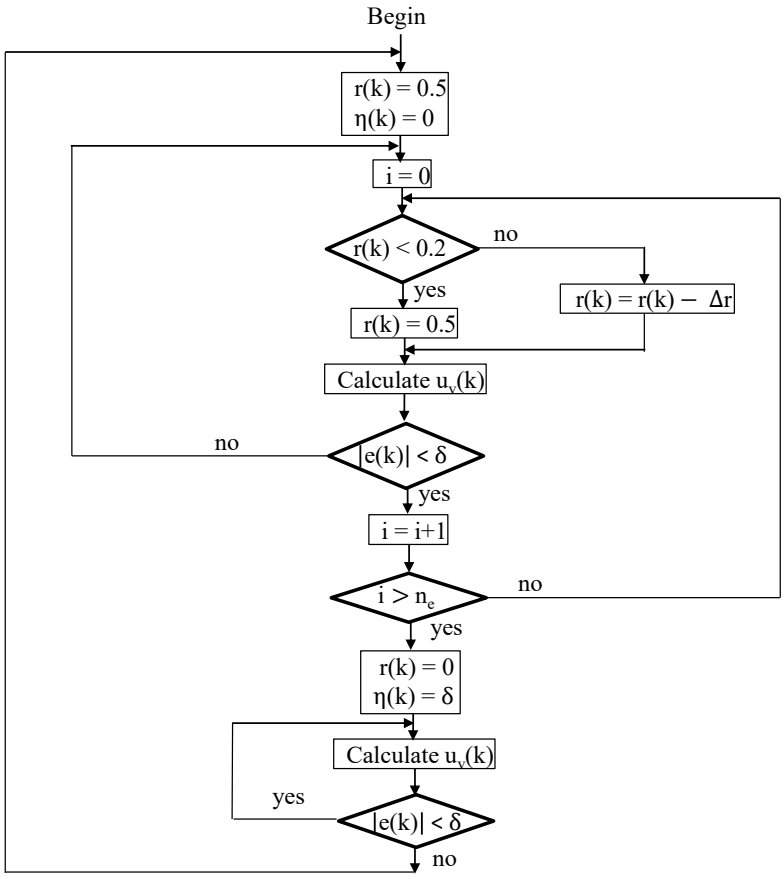


**Figure 7.2:** Flowchart of the proposed algorithm

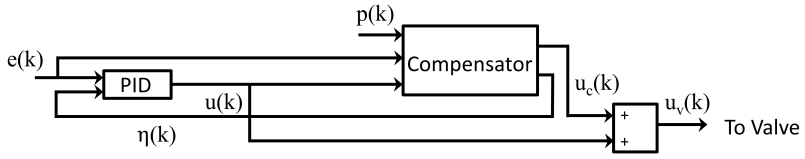


**Figure 7.3:** Integration of the proposed compensation scheme with PID block.

The use of this variable compensating signal to the pulses is threefold: the valve travel is reduced; the possibility of a change in the valve position when the pulses are ceased is reduced, since the amplitude is decreasing; since stiction have different values for different valve positions, the search provided by the ramp will result in the proper amplitude of the pulses for each situation.

The integration of the proposed algorithm with the existing PID controller is shown in Figure 7.3. The compensator generates signal $u_c(k)$ using the error $e(k)$, the controller output $u(k)$ and the pulse $p(k)$. The calculation of the signal applied to the valve $u_v(k)$ is

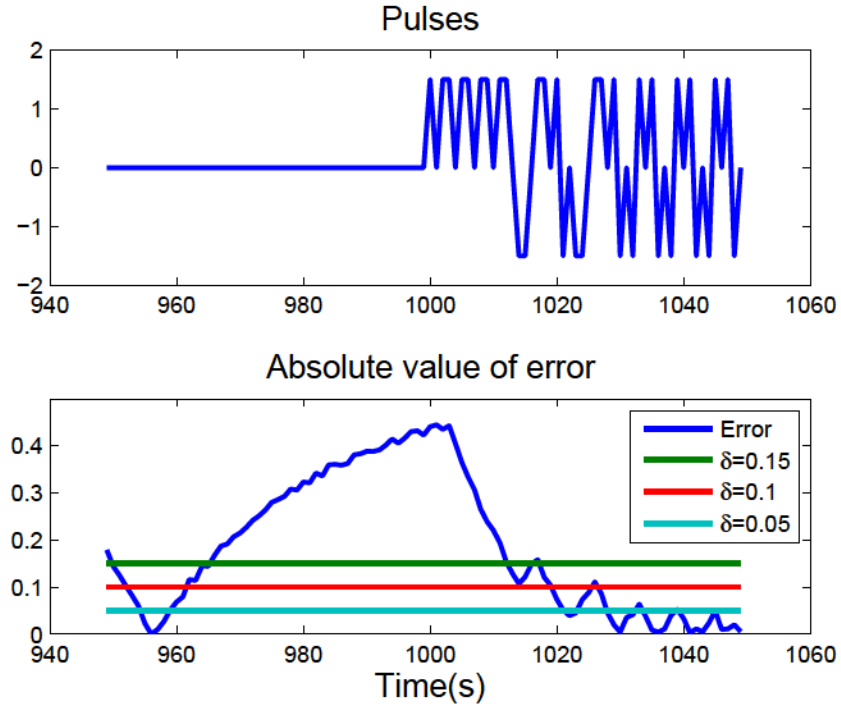


**Figure 7.4:** Choice of limit ($\delta$) for the error $e(k)$.

obtained summing $u(k)$ and $u_c(k)$. The results with different pulses $p(k)$ such as knocker, constant value and rectangular will be analyzed via simulations in Section 7.2.

### 7.1.1 Parameters of the Algorithm

The proposed algorithm requires only three parameters to be obtained from the control loop with the sticky valve. An estimation $S$ of the stiction can be approximated by the amplitude of controller output $u(k)$, that can be obtained observing the trends on a supervisory system. The sample time $T_s$ is easily obtained. Finally, an estimate of the time-constant of the control loop is required. The operators have no difficulty to provide such estimation. For example, flow loops have in general time-constants around 2 s. With these parameters, one can calculate $n_e$, $n_r$ and $\Delta r$, using equations 7.6, 7.5 and 7.4 respectively to run the algorithm. The user should also specify the limit for the error between the setpoint (SP) and the process-variable (PV) $\delta$. This value can also be obtained based on the error in the limit cycles. Another way to specify this limit is to apply pulses with constant amplitude of approximately $0.5S$ and to measure the achievable error, like shown in Figure 7.4.

If knocker pulses are used, the parameters related to the duty cycle of the pulses should be chosen. For constant pulses and a for square wave no extra parameters are required.

## 7.2 Simulation

The friction model used for all the simulations done in this chapter was first proposed by Choudhury et al. (2005, 2008b) and then improved by Xie (2013). This model uses only two parameters (dead-band $S$ and slip-jump $J$), and requires low computational effort for simulations. Moreover, it was demonstrated in Xie (2013) that it can properly represent the behavior of stiction. The simulation was performed with a process modeled by transfer function

$$G(s) = \frac{1}{10s + 1} \tag{7.7}$$

using the general controller settings as

$$C(s) = K_c + \frac{1}{T_i s}. \tag{7.8}$$

where, the proportional gain ($K_c$) was used as 0.47 and integral time-constant ($T_i$) was used as 6.25 in all these simulations.

The parameters for the stiction model were $S = 2$ and $J = 1$, with sample time $T_s = 1$ s. A uniform random noise with amplitude 0.01 was added to system output. A knocker compensator with pulse width of $2T_s$ and time between pulses of $3T_s$ was used. The limit for the error was $\delta = 0.2$, obtained via simulation using only PI controller. Parameters $n_e = 10$ and $n_r = 40$ were calculated using equations 7.6 and 7.5 respectively.

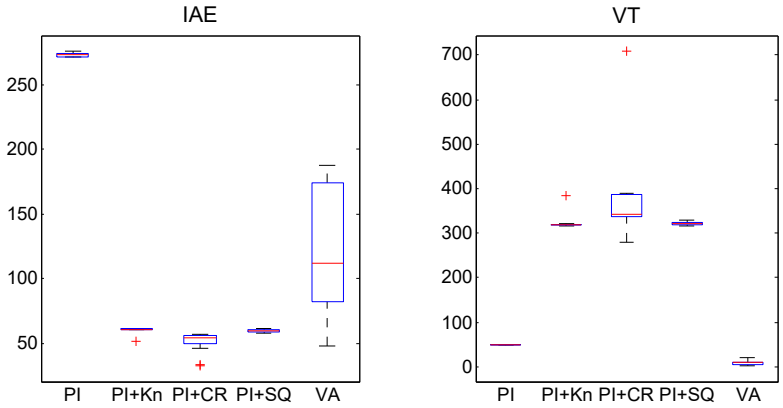


**Figure 7.5:** Comparison among different stiction compensation schemes.

Figure 7.5 shows the box plots for five different strategies to handle control valve stiction:

- PI Controller (PI)

- PI Controller + pulses from Knocker method (PI+Kn)
- PI Controller + constant pulses (Constant Reinforcement method) (PI+CR)
- PI Controller + the square wave pulses (PI+SQ)
- Proposed method with pulses with variable amplitude (VA)

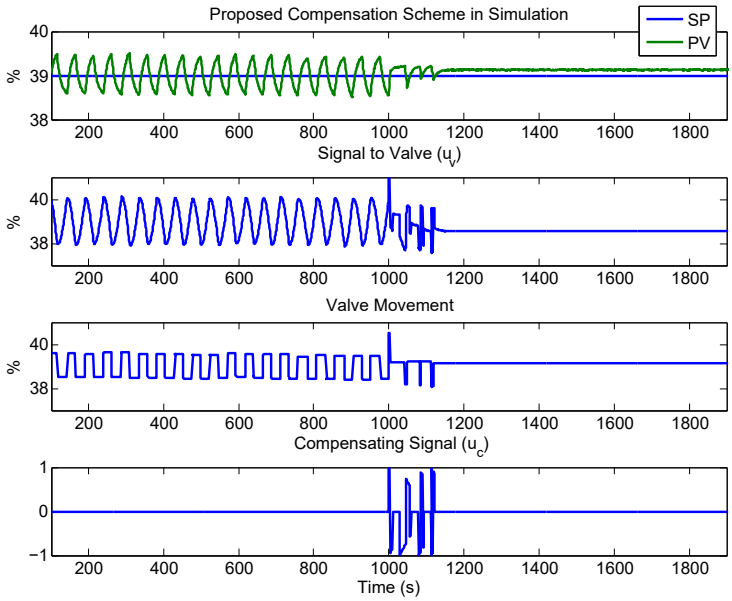


**Figure 7.6:** Application of the method to a simulated sticky valve.

The simulation for each method was repeated 20 times, adding a random value between 0 and 10 to a fixed setpoint of 30. This figure shows that the PI+CR produces the best IAE with the price of a high valve travel. This is because the PI+CR method employs the highest energy to combat stiction. It also shows that when only the PI controller was active, valve travel was less than PI+Kn, PI+CR and PI+SQ methods but it had the highest IAE. Valve travel using PI was small since limit cycles have frequency smaller than the switching produced by the pulses. PI+Kn and PI+SQ are quite comparable in their performances regarding both IAE and VT. However, the proposed method has IAE much lower error than the PI method and has the lowest valve travel. So, the proposed method presents a good trade-off for both IAE and VT indexes. A larger variation on the indexes IAE and VT is expected in this method since the time required to cease the pulses is random. It is worth to say that a small offset between PV and SP is usually tolerated in the industry if there is no oscillation.

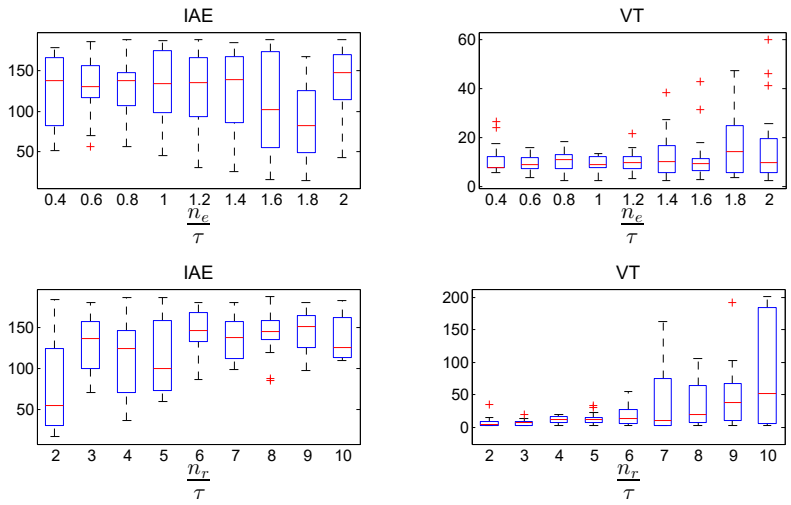


**Figure 7.7:** Simulations for choosing parameters $n_e$ and $n_r$.

One of these simulations is shown in Figure 7.6. The panels show all signals involved with the control loop, including the valve movement which comes from friction model. At instant 1000 s the compensation starts and at 1119 s the error becomes smaller than the threshold. The indexes VT and IAE are measured from the beginning of the compensation until the end of the simulation. The amplitude of controller signal is about 2% ($S = 2$) and the amplitude of the pulses start with value 1. A simulation was performed to show the effect of variations for 9 different values of parameters $n_e$ and $n_r$ on IAE and VT. The values of these parameters were selected around those provided by the respective equations, 7.5 and 7.6. For each set of parameters, the simulation was repeated for 20 different random setpoints. The results in Figure 7.7 show that variations in IAE and VT are small. Therefore, one can see that $\frac{n_r}{\tau} = 4$ and $\frac{n_e}{\tau} = 1$ (for $T_s = 1$) are good choices for these parameters. Variations around them caused by approximations have a negligible effect on IAE and VT.

## 7.3 Real Application and results

The proposed method was applied to a pilot plant (Figure 7.8), for both flow and level control loops, selected by the switch $SW$. The position of valve 2 and the pump rotation are adjusted for the plant to operate in the desired condition. Valve 1 is used as actuator both for flow and level control. The flow and level transmitters and the I/P transducer work with 4 to 20 mA signals. A programmable industrial controller (PLC) from Freelance 2000 from ABB was used to control and monitor the signals and to implement the proposed

compensator as well. Valve 1 is a 2 inch globe pneumatic control valve with positioner (Figure 7.9). The position signal is measured for the purpose of position control but is not available to the user, since it brings an extra cost to the valve. Friction was increased by tightening the valve stem packing. The presence of the positioner is a usual situation in the industrial environment. However, the test with stiction compensation is usually done in valves without positioner. It will be shown in this application that any method to combat stiction will fail if no special attention is directed to the role of the positioner.

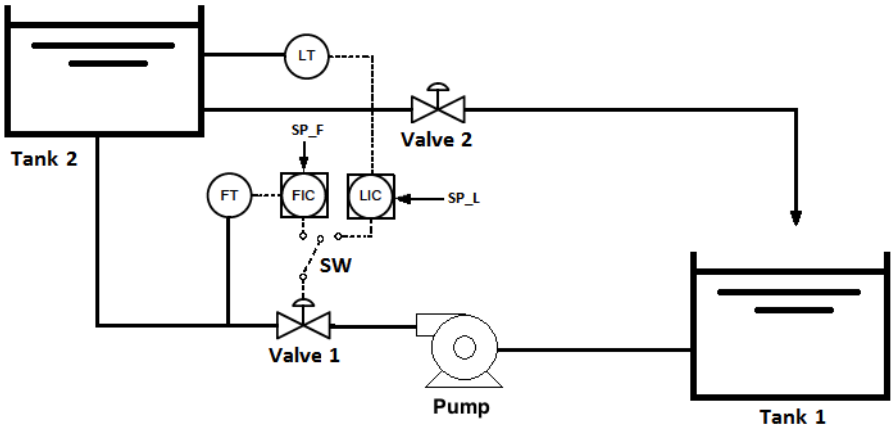


**Figure 7.8:** Diagram of flow and level control loops for real application.

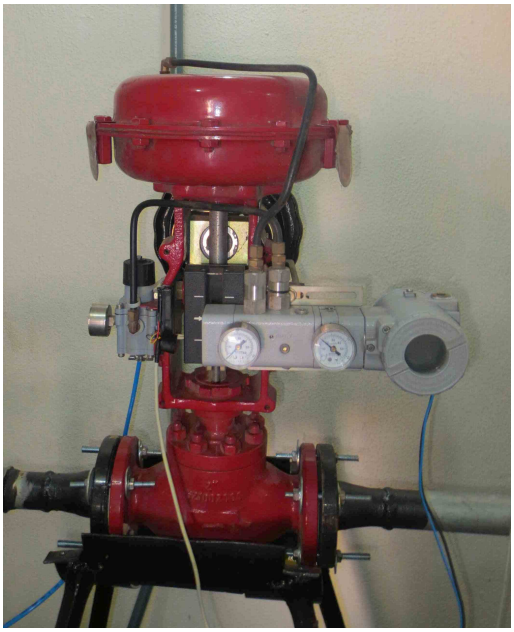

**Figure 7.9:** Sticky control valve with positioner.

### 7.3.1 Compensator Code

The compensator was programmed in structured text, which is one of the five languages supported by the IEC 61131-3 standard, designed for programmable logic controllers. This choice greatly simplifies transferring the program from one PLC to another, even from a different manufacturer.

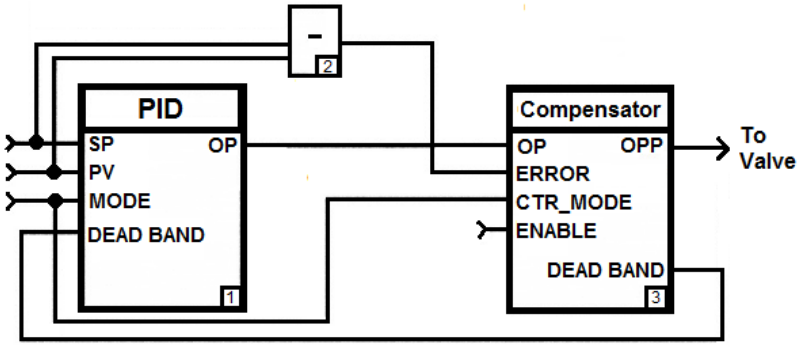


**Figure 7.10:** PID and compensator programmed under Functional Block Diagram language.

The compensator code is called in the context of the main code programmed in Functional Block Diagram language, shown in Figure 7.10. The PID controller is a block available on PLCs and is connected with the compensator block as shown in Figure 7.10. One can compare Figure 7.10 with Figure 7.3 (connection between PID and compensator), which describes the flowchart of the proposed compensator. If the PID is set to manual mode, the pulses are ceased through $CTR\_MODE$ in compensator block. The same happens when the compensator is disabled using the input $ENABLE$. The compensator sends to the PID a dead-band value to be used for integral action when the error satisfies the specified limit. This input is commonly used to reduce the activation of the output, thus reducing the wear of the actuator. In order to work with the compensator block, the PID block should have the inputs shown: SP, PV and MODE (they are usual in commercial PID blocks). Therefore, if these inputs are available on PID block, its interface with the proposed compensator can be accomplished.

### 7.3.2 Application on Flow Loop of the Pilot Plant

The switch SW was connected to the flow loop (Figure 7.8) using a PI controller with sample time $T_s = 0.25$s. The value of 2s was used for the time-constant $\tau$ of the loop. Initially, the setpoint for the flow loop was fixed at 11 $l/min$. A sustained limit cycle appears (Figure 7.11), and the amplitude of the controller output ($u(k)$) is around 1.8%

(bottom panel of Figure 7.11). So, the approximate value of stiction parameter $S$ is 1.8. One can see that as soon as the control signal overcomes stiction, the flow jumps to a new value, caused by the jump of the valve to a new position. With the values of $\tau$ and $T_s$, the parameters $n_e$ and $n_r$ were calculated as 8 and 32 respectively, using equations 7.6 and 7.5. The limit for error ($\delta$) was specified as 0.3, by observation of PV and SP signals in top panel of Figure 7.11.

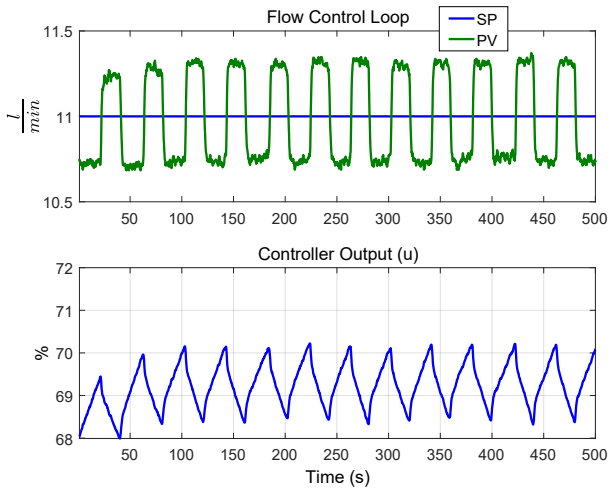


**Figure 7.11:** Limit cycles due to stiction in the flow loop.

The bottom panel of Figure 7.12 shows the applied compensating signal calculated according to the proposed algorithm. The compensation started on time instant 500 s. The pulses $p(k)$ come from the knocker signal with width of $2T_s$, time between pulses of $3T_s$ and amplitude 0.9. In the bottom panel one can see the pulses added to controller signal. The amplitude of the compensating signal $u_c(k)$ is decreased during 8s, unless the error becomes smaller than $\delta = 0.3$ during 2 s. This condition was satisfied at the instant 575 s: the pulses were ceased, the dead-band on integral action was activated and the limit cycle disappeared. This condition was clearly satisfied earlier, but after turning the compensation off the error became larger than the limit and the compensation was resumed.

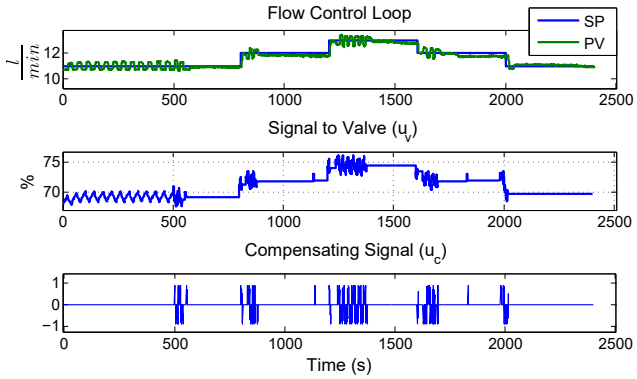


**Figure 7.12:** Effect of the proposed compensation scheme on flow loop.

Figure 7.13 is a zoomed-in version of Figure 7.12 which gives more details of the algorithm. This figure shows that when error became larger than the specified limit (0.3 $l/min$) at 830 s, both the integral action of the PI controller and the pulses were resumed. The amplitude of the pulses is initially 0.9 ($0.5S$) and after 8 s is reduced to 0.36 ($0.2S$). The pulses are added in the same direction of controller output. The compensation succeeds at 878 s. However, the compensation stops two times before this instant (866.8 s and 868.8 s), since the condition was satisfied but the dynamics of the flow loop caused error to increase more than the specified limit. The choice of parameter $n_e$ aims to reduce the chance of this situation to happen.

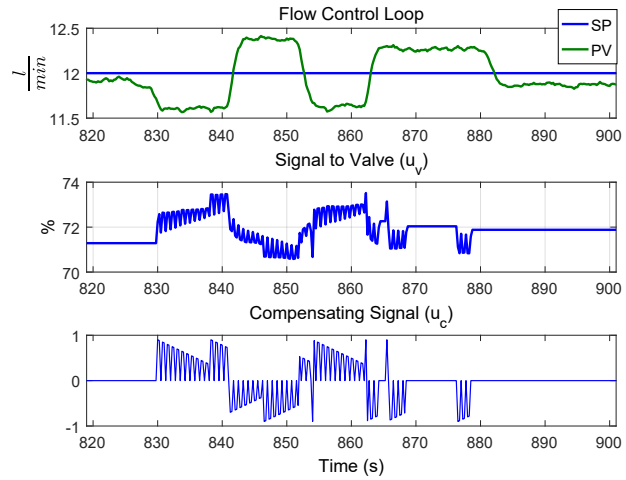


**Figure 7.13:** This figure is a zoomed-in version of Figure 7.12.

To show the ability of the proposed compensation to track dynamic setpoint changes, the setpoint was increased to 12, 13 and then decreased to 12 and 11 (see Figure 7.12). No

change was required in the parameters of the compensator during these setpoint changes. The compensation started and stopped automatically, bringing the error within the desired limit all the times. Both the time required to achieve the desired error and the steady state error change every time the compensation scheme works, which is expected due to random behavior of signals.

The application of the proposed compensation to a level loop, which has slower dynamics is shown in the next subsection.

### 7.3.3 Application on the Level Loop of the Pilot Plant

For the tests with a level loop, switch $SW$ was changed to level control and a PI controller with the sampling-time of 1 s was used. The time-constant was chosen 80 s, which is an approximate value for this loop. With a setpoint 35%, the loop was closed, and the effect of stiction appeared, as shown in Figure 7.14. The controller output was oscillating between 68.95% and 71.05%, which gives an approximate value of $S = 2.10$ for stiction. Form the observation of the error, its limit was chosen 0.3. This information is gathered during the first 1000 s of the test, and are used to calculate $n_e = 80$ and $n_r = 320$. The knocker parameters to generate the pulse for the compensating signal were: time between pulses as $4T_s$, the width of pulse as $2T_s$ and amplitude as 1.05 (approximately $0.5S$).

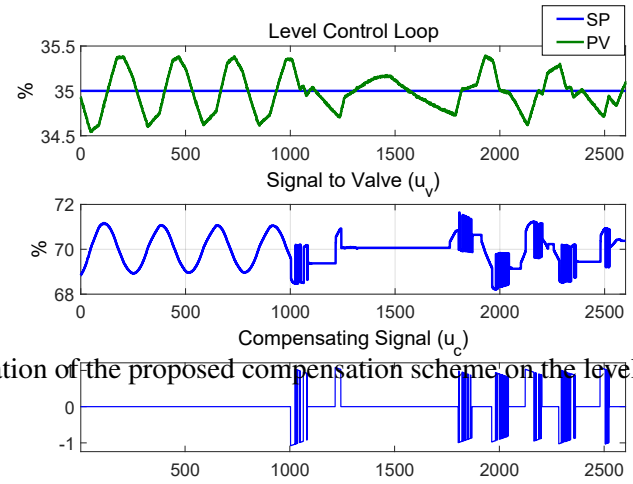


**Figure 7.14:** Application of the proposed compensation scheme on the level loop of the pilot plant.

The compensation started at instant 1003 s (Figure 7.14). The amplitude of oscillation was reduced, and the error as well, but the level didn't reach a steady state. The change in the level resumed the compensation. The reason for this misbehavior is the activity of the valve positioner, that has a control loop that tries to make the position error equal to zero, using the integral action of the controller. This effect didn't appear on flow loop because of its fast dynamics. The remedy is to detune the integral action, as discussed detailed in Mohammad and Huang (2012).

The integral time-constant of the positioner was changed from 4 to 20 s, using an asset management software to configure the positioner. It is important to highlight that any method to compensate stiction that allows a small error between SP and PV would not be able to stop the valve movement in such situation. On the other hand, they can get the benefit of this detuning strategy.

Figure 7.15 shows the result with the proposed compensation scheme on the same level loop after detuning the integral action of the valve positioner. The first 1000 s are used to check the error and controller output signal.

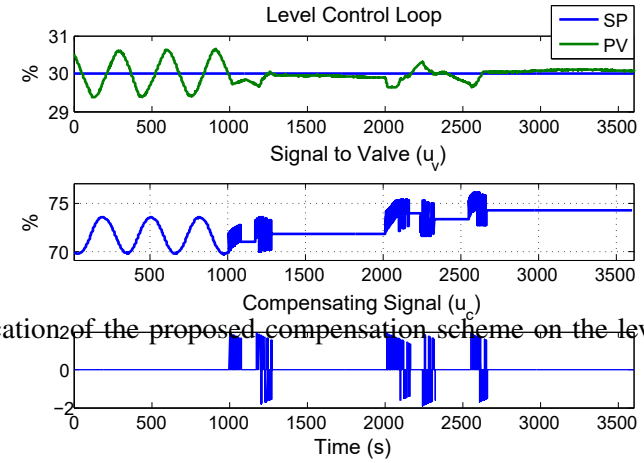


**Figure 7.15:** Application of the proposed compensation scheme on the level loop after detuning valve positioner.

Although the parameters of the level controller remained the same, one can see that the amplitude of controller signal increased to 3.8% (as seen from Figure 7.15). This is the effect of detuning the controller of the positioner. Hence, the estimated stiction is updated and so is the amplitude of the pulses, which start now with the value 1.9 ($0.5S$). The compensation started at 1004 s and for the first time it could bring error within the limit at 1081 s. It was resumed again shortly when the error went beyond the limit at 1176 s.

The compensation restarted automatically only when a disturbance was introduced in the loop by reducing its steady flow-rate from 15.3 $l/min$ by reducing the pump speed in 6%. This deliberate disturbance was applied at 2000 s and the error started increasing. The compensation was resumed at 2015 s when the error was greater than $\delta$. The algorithm could handle this disturbance successfully. It is to be noted that the author could not apply a dead-band for the integral part of the valve positioner but could just detune it. That is why a small change in level is still seen, though the PI controller was not active. Anyway, the valve is not moving most of the time.

## 7.4 Concluding Remarks

In this work, a new method for stiction compensation using pulses with variable amplitude has been proposed. The method was applied to a simulation example and compared to commonly used methods, producing better results when considering the indexes IAE and valve travel. Simulations were also performed to illustrate the flexibility for selecting the required parameters. The application to an industrial controller with a real sticky valve both for a flow loop and a level loop was successful, meaning that control loops with slow and fast dynamics can be considered. The effect of valve positioner, very common in the industry, was analyzed, and it is shown that stiction compensation methods can fail if the suggested guidelines are not obeyed. Its implementation in a commercial PLC and continuous operation for different setpoints and under disturbances confirms its simplicity and applicability in industrial environments.

# Chapter 8

# Final Conclusions

## 8.1 Contributions of this Thesis

The main contributions of this thesis are:

- A novel data-driven, non-parametric, online change detection method has been proposed. The change detection algorithm is based on the optimal Kantorovich distance algorithm. The method was successfully applied to the benchmark Tennessee Eastman process and on a real pipeline data in an online mode.

- A particle filter based novel leak detection algorithm was proposed. The method was successful in detecting and localizing leak in a simulated setting. However, due to uncertainties in the model and insufficient amount of data, the particle filter based leak detection and localization system (LDS) was not successful in dealing with real industrial data.

- To overcome the limitations of model-based LDS, a data-driven LDS was proposed based on the Kantorovich distance algorithm. The data-driven LDS was successfully applied to industrial pipeline data at 'Suncor Pipelines' during the annual regulatory leak tests in real-time for a leak as small as 1%. The detection of this the leak was much faster than the existing commercial method in use at Suncor pipelines. Leak isolation (localization) efficiency of the proposed data-driven LDS was also comparable to the existing commercial method.

- Control valves are the most commonly used actuators in the process and pipeline industries. As such non-linearities in control valve, especially stiction, have a disastrous effect on normal process operation. A novel data-driven stiction detection method has been proposed based on the Kantorovich distance. The only solution to this physical problem is to repair or replace the valve. A novel stiction compensation

method has been proposed to alleviate the effect of stiction till the next repair cycle during process shutdown. In this method, the compensating signal is proportional to the error between the setpoint and process-variable. The method was applied successfully to a real sticky valve at the CPC laboratory of University of Alberta for both flow and level control loop.

- The control valve stiction compensation method described in chapter 6 has one disadvantage. Its performance greatly reduces, if the error does not come within the tolerance limit with the amplitude of the compensating signal less than or equal to half of the stiction parameter $S$. In such situations, the error keeps increasing with the increase of compensating signal and there is no convergence. To overcome this limitation, a new method has been proposed, where the compensating signal is ramp modulated and the amplitude of the compensating signal traverses between $0.2S$ to $0.8S$. Thus, when the error comes within the specified limit, both the compensating signal and integral action of PI controller are suspended till the next case when the error crosses the limit. This method is also capable of handling variability in stiction parameter. The method was successfully applied to a pilot plant equipped with an industrial scale control valve including a built-in valve positioner both for the flow and level loops.

### 8.1.1 Publications

- A manuscript for the change detection algorithm based on Kantorovich distance concept work has been accepted for the $10^{th}$ IFAC symposium on Advanced Control of Chemical Processes (ADCHEM) to held on 2018 at Shenyang, China. The proposed title for this conference paper is 'Change Point Detection using the Kantorovich Distance Algorithm'.

- One conference paper has already been published with initial results from the particle filter based leak detection method titled as 'Pipeline Leak Detection Using Particle Filters' Arifin et al. (2015) in the $9^{th}$ IFAC symposium on Advanced Control of Chemical Processes (ADCHEM) held on 2015 at Whistler, British Columbia, Canada. The article is available online at IFAC-PapersOnline 48-8 (2015) pp. 076-081.

- The data-driven leak detection method has been published in the journal of 'Computers & Chemical Engineering' titled as 'A Novel Data-Driven Leak Detection and

Localization Algorithm using the Kantorovich Distance',2018, Volume 108, pages 300-313 (Arifin et al., 2018a).

- An abridged version of the proposed stiction compensation method where compensating signal is proportional to the absolute error has already been published titled as 'A Model Free Approach for Online Stiction Compensation' Arifin et al. (2014) in the $19^{th}$ IFAC World Congress (2014) held in Cape Town, South Africa. The article is available online at IFAC Proceedings Volumes, 47-3 (2014) pp. 5957-5962.

- A slightly modified version of the variable amplitude stiction compensation has been published in the journal of ISA Transactions® as 'Actuator Stiction Compensation via Variable Amplitude Pulses', 2018, Volume 73, pages 239-248 (Arifin et al., 2018b).

## 8.2 Industrial Application

The data-driven leak detection method was successfully applied to the industrial pipeline in real-time during the annual regulatory leak tests at Suncor Pipelines. The method was successful in detecting and localizing a leak as small as 1% with comparable leak localization efficiency and much faster detection time in comparison to the existing commercial method.

The variable amplitude control valve stiction compensation method was applied to a pilot plant equipped with an industrial size physical sticky valve with a valve stem positioner. The method was successful not only in reducing the limit cycle but also in reducing the valve movement in comparison with the knocker and CR methods. The method is also capable of handling uncertainty in stiction measurement, dynamic setpoint tracking and disturbance rejection.

Another important feature of the proposed method is that it was implemented on an industrial DCS system using IEC 61131-3 standard, which makes its industrial application on other industrial DCS systems quite feasible.

### 8.2.1 Limitations

The proposed data-driven leak detection algorithm has the following limitations :

- Detection of 1% leak is dependent on the accuracy of flow-meters of injection and delivery nodes. If summation of uncertainties of the injection and delivery flow-meters are less than 1% only then 1% leak is detectable. Suncor did perform flow-meter proving before the regulatory leak test for the purpose of 1% leak detection.

- Leak localization is not very precise with the proposed algorithm. It can localize leak at the accuracy of half of the distance between two consecutive pressure transmitters. So, leak localization is dependent on the location of the pressure transmitters. Accuracy of leak localization increases with the increment of installed pressure transmitters.

The proposed variable amplitude control valve stiction compensation method has the following limitations :

- The time required to cancel out the effect of the limit cycle for the proposed variable amplitude control valve stiction compensation method is random. Sometimes it may take longer to diminish the effect of limit cycle which may increase the amount of valve reversals.

- If the valve is equipped with built-in valve positioner, the integral part of the valve positioner needs to be detuned to get expected performance from the compensation algorithm. This is also true for all the available stiction compensation algorithms.

## 8.3 Directions for Future Work

The Kantorovich distance based change detection method can be applied in the field of data segmentation, alarm rationalization, batch tracking and many other applications.

For the proposed leak detection method, dynamic thresholding can be implemented in future to reduce false alarms. The 3 thresholds, those are needed to be set are currently provided as fixed values. So, if these thresholds can be updated dynamically there will be fewer false alarms and the detection of smaller and larger leaks will be more reliable. More complex situation with leak detection such as multiple injection and deliveries of fluid, batch processes, variable frequency drive (VFD) pumps, pressure control valves (PCV), column separation can be addressed in future research.

The model-based LDS can be re-investigated since a properly working and properly tuned model can predict the hydraulic nature of the pipeline which may provide important insight into the pipeline. Tuning the model properly is a challenging work.

The choice for error limit in the proposed stiction compensation method is user-defined until this point. This can be automated. The estimation of stiction can be automated too. Then the time-constant and the sampling-time of the control loop will be the only parameters to be provided by the operator.

# Bibliography


Adams, R. P., MacKay, D. J., 2007. Bayesian online changepoint detection. arXiv preprint arXiv:0710.3742.

Adnan, N. A., Izadi, I., Chen, T., 2011. On expected detection delays for alarm systems with deadbands and delay-timers. Journal of Process Control 21 (9), 1318–1331.

Al-Khomairi, A. M., 1995. Improving leak detectability in long liquids pipelines. Ph.D. thesis.

Al-Rafai, W., Barnes, R., 1999. Underlying the performance of real-time software-based pipeline leak-detection systems. Pipes and Pipelines International 44 (6), 44–51.

Applegate, D., Dasu, T., Krishnan, S., Urbanek, S., 2011. Unsupervised clustering of multidimensional distributions using earth mover distance. In: Proceedings of the $17^{th}$ ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 636–644.

Arifin, B. M. S., Li, Z., Shah, S. L., 2015. Pipeline leak detection using particle filters. IFAC-PapersOnLine 48 (8), 76–81.

Arifin, B. M. S., Li, Z., Shah, S. L., Meyer, G. A., and Colin, A., 2018a. A novel data-driven leak detection and localization algorithm using the kantorovich distance. Computers & Chemical Engineering 108, 300 – 313.

Arifin, B. M. S., Munaro, C. J., Choudhury, M. S., Shah, S. L., 2014. A model free approach for online stiction compensation. Presented in the $19^{th}$ IFAC World Congress held in Cape Town, South Africa, IFAC Proceedings Volumes, 47-3, 5957–5962.

Arifin, B. M. S., Munaro, C. J., Angarita, O. F. B., Cypriano, M. V. G., and Shah, S. L., 2018b. Actuator stiction compensation via variable amplitude pulses. ISA Transactions 73, 239 – 248.

Armstrong-Hëlouvry, B., Dupont, P., Wit, C. C. D., 1994. A survey of models, analysis tools and compensation methods for the control of machines with friction. Automatica 30 (7), 1083 – 1138.

Arulampalam, M. S., Maskell, S., Gordon, N., Clapp, T., 2002. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. Signal Processing, IEEE Transactions on 50 (2), 174–188.

Baillieul, J., Samad, T., 2015. Encyclopedia of systems and control. Springer Publishing Company, Incorporated, 417-422.

Balakrishnapillai Chitralekha, S., 2011. Computational tools for soft sensing and state estimation. Ph.D. thesis, copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated on - 2016-03-10.

Basseville, M., Benveniste, A., 1983. Desgin and comparative study of some sequential jump detection algorithms for digital signals. Acoustics, Speech and Signal Processing, IEEE Transactions on 31 (3), 521–535.

Basseville, M., Nikiforov, I. V., et al., 1993. Detection of abrupt changes: theory and application. Vol. 104. Prentice Hall Englewood Cliffs.

Bathelt, A., Ricker, N. L., Jelali, M., 2015. Revision of the tennessee eastman process model. IFAC-PapersOnLine 48 (8), 309–314.

Billmann, L., Isermann, R., 1987. Leak detection methods for pipelines. Automatica 23 (3), 381–385.

Brasio, A. S., Romanenko, A., Fernandes, N. C., 2014. Modeling, detection and quantification, and compensation of stiction in control loops: The state of the art. Industrial & Engineering Chemistry Research 53 (39), 15020–15040.

Chetouani, Y., 2012. Change detection in a distillation column based on the generalized likelihood ratio approach. Journal of Chemical Engineering & Process Technology 2011.

Choudhury, M. A. A. S., Jain, M., Shah, S. L., 2008a. Stiction–definition, modelling, detection and quantification. Journal of Process Control 18 (3), 232–243.

Choudhury, M. A. A. S., Shah, S. L., Thornhill, N. F., 2008b. Diagnosis of process nonlinearities and valve stiction: data driven approaches. Springer.

Choudhury, M. A. A. S., Thornhill, N. F., Shah, S. L., 2005. Modelling valve stiction. Control Engineering Practice 13 (5), 641–658.

Cuadros, M., Munaro, C. J., Munareto, S., 2012. Novel model-free approach for stiction compensation in control valves. Industrial and Engineering Chemistry Research 51 (25), 8465–8476.

Daugherty, R. L., Franzini, J. B., 1977. Fluid mechanics, with engineering applications. McGraw-Hill NY, Chapter 12, 393-396.

Desborough, L., Miller, R., 2002. Increasing customer value of industrial control performance monitoring-honeywell's experience. In: AIChE symposium series. New York; American Institute of Chemical Engineers; 1998, 169–189.

Di Blasi, M., Muravchik, C., 2009. Leak detection in a pipeline using modified line volume balance and sequential probability tests. Journal of pressure vessel technology 131 (2), 021701–021701-7.

Downs, J. J., Vogel, E. F., 1993. A plant-wide industrial process control problem. Computers & chemical engineering 17 (3), 245–255.

Durand, H., Christofides, P. D., 2016a. Actuator stiction compensation via model predictive control for nonlinear processes. AIChE Journal 62, 2004–2023.

Durand, H., Christofides, P. D., 2016b. Stiction compensation via model predictive control. In: American Control Conference (ACC). Boston, Massachusetts, 4488–4493.

EEMUA, 2007. Alarm systems: A guide to design, management and procurement.

Entech, 1998. Control valve dynamic specification.

Fletcher, R., Chandrasekaran, M., 2008. Smartball¢: A new approach in pipeline leak detection. In: 2008 $7^{th}$ International Pipeline Conference. American Society of Mechanical Engineers, 117–133.

Fukushima, K., Maeshima, R., Kinoshita, A., Shiraishi, H., Koshijima, I., 2000. Gas pipeline leak detection system using the online simulation method. Computers & chemical engineering 24 (2), 453–456.

Gao, Y., Wang, X., Wang, Z., Zhao, L., 2016. Fault detection in time-varying chemical process through incremental principal component analysis. Chemometrics and Intelligent Laboratory Systems 158, 102–116.

Ge, C., Wang, G., Ye, H., 2008. Analysis of the smallest detectable leakage flow rate of negative pressure wave-based leak detection systems for liquid pipelines. Computers & Chemical Engineering 32 (8), 1669–1680.

Geiger, G., 2006. State-of-the-art in leak detection and localization. Oil Gas European Magazine 32 (4), 193–198.

Gerry, J., Ruel, M., 2001. How to measure and combat valve stiction online. In: ISA International Fall Conference, Houston, TX.

Gröwe-Kuska, N., Heitsch, H., Römisch, W., 2003. Scenario reduction and scenario tree construction for power management problems. In: Power Tech Conference Proceedings, 2003 IEEE Bologna. Vol. 3. IEEE, 7–pp.

Hägglund, T., 2002. A friction compensator for pneumatic control valves. Journal of Process Control 12 (8), 897–904.

Horch, A., 2000. Condition monitoring of control loops. Ph.D. thesis, KTH.

Hwang, I., Kim, S., Kim, Y., Seah, C. E., 2010. A survey of fault detection, isolation, and reconfiguration methods. IEEE Transactions on Control Systems Technology 18 (3), 636–653.

ISA, 2009. Management of alarm systems for the process industries.

ISA Subcommittee, 1979. Process instrumentation terminology technical report ansi/isa-s51. 1-1979. Instrument Society of America.

Isermann, R., 2005. Model-based fault-detection and diagnosis–status and applications. Annual Reviews in control 29 (1), 71–85.

Izadi, I., Shah, S., Shook, D., Chen, T., 2009a. An introduction to alarm analysis and design. In: Proceedings of $7^{th}$ IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes., 645–650.

Izadi, I., Shah, S. L., Chen, T., 2010. Effective resource utilization for alarm management. In: Decision and Control (CDC), 2010 $49^{th}$ IEEE Conference on. IEEE, 6803–6808.

Izadi, I., Shah, S. L., Shook, D. S., Kondaveeti, S. R., Chen, T., 2009b. A framework for optimal design of alarm systems. In: Proceedings of $7^{th}$ IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes., 651–656.

Kaijser, T., 1998. Computing the kantorovich distance for images. Journal of Mathematical Imaging and Vision 9 (2), 173–191.

Kawahara, Y., Sugiyama, M., 2012. Sequential change-point detection based on direct density-ratio estimation. Statistical Analysis and Data Mining 5 (2), 114–127.

Kawahara, Y., Yairi, T., Machida, K., 2007. Change-point detection in time-series data based on subspace identification. In: $7^{th}$ IEEE International Conference on Data Mining (ICDM 2007). IEEE, 559–564.

Kayihan, A., Doyle, F. J., 2000. Friction compensation for a process control valve. Control Engineering Practice 8 (7), 799–812.

Kivinen, J., Smola, A. J., Williamson, R. C., 2004. Online learning with kernels. Signal Processing, IEEE Transactions on 52 (8), 2165–2176.

La Torre, D., Mendivil, F., 2015. The monge–kantorovich metric on multimeasures and self–similar multimeasures. Set-Valued and Variational Analysis 23 (2), 319–331.

Levy, B. C., 2008. Principles of signal detection and parameter estimation. Springer Science & Business Media.

Li, Z., Li, Z., 2016. Linear programming-based scenario reduction using transportation distance. Computers & Chemical Engineering 88, 50–58.

Liu, M., Zang, S., Zhou, D., 2005. Fast leak detection and location of gas pipelines based on an adaptive particle filter. International Journal of Applied Mathematics and Computer Science 15 (4), 541.

Liu, S., Yamada, M., Collier, N., Sugiyama, M., 2013. Change-point detection in time-series data by relative density-ratio estimation. Neural Networks 43, 72–83.

Lv, C., Wang, H., Cao, D., 2017. High-precision hydraulic pressure control based on linear pressure-drop modulation in valve critical equilibrium state. IEEE Transactions on Industrial Electronics PP (99), 1–1.

Martinez-Guerra, R., Mata-Machuca, J. L., 2016. Fault detection and diagnosis in nonlinear systems. Springer.

Mishra, P., Kumar, V., Rana, K., 2014. A novel intelligent controller for combating stiction in pneumatic control valves. Control Engineering Practice 33, 94–104.

Mishra, P., Kumar, V., Rana, K., 2015. An online tuned novel nonlinear pi controller for stiction compensation in pneumatic control valves. ISA Transactions 58, 434–445.

Modisette, J. P., et al., 2013. State estimation of pipeline models using the ensemble kalman filter. In: PSIG Annual Meeting. Pipeline Simulation Interest Group.

Mohammad, M. A., Huang, B., 2012. Compensation of control valve stiction through controller tuning. Journal of Process Control 22 (9), 1800–1819.

Munaro, C. J., de Castro, G. B., da Silva, F. A., Angarita, O. F., Cypriano, M. V. G., 2016. Reducing wear of sticky pneumatic control valves using compensation pulses with variable amplitude. IFAC-PapersOnLine 49 (7), 389–393.

Murvay, P.-S., Silea, I., 2012. A survey on gas leak detection and localization techniques. Journal of Loss Prevention in the Process Industries 25 (6), 966–973.

Nagel, T., Zhang, C., Liu, S., 2012. Kalman filter based leak localization applied to pneumatic systems. In: Control Automation Robotics & Vision (ICARCV), $12^{th}$ International Conference on. IEEE, 1777–1782.

Nielsen, M. A., 2015. Neural networks and deep learning. Determination Press USA, Chapter 1.

Olsson, H., 1996. Control systems with friction. PhD Theses.

Ostapkowicz, P., 2014. Leakage detection from liquid transmission pipelines using improved pressure wave technique. Eksploatacja i Niezawodność - Maintenance and Reliability 16(1), 9–16.

Raghavan, H., 2004. Quantitative approaches for fault detection and diagnosis in process industries. Ph.D. thesis, copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-03-14.

Reddy, H. P., Narasimhan, S., Bhallamudi, S. M., Bairagi, S., 2011. Leak detection in gas pipeline networks using an efficient state estimator. part-i: Theory and simulations. Computers & chemical engineering 35 (4), 651–661.

Ristic, B., Arulampalam, S., Gordon, N. J., 2004. Beyond the Kalman filter: Particle filters for tracking applications. Artech House.

Ruel, M., 2000. Stiction: The hidden menace. Control Magazine 13 (11), 1–11.

Seborg, D., Mellichamp, D., Edgar, T., 2006. Process Dynamics and Control, 2nd ED. Wiley India Pvt. Limited.

Silva, R. A., Buiatti, C. M., Cruz, S. L., Pereira, J. A., 1996. Pressure wave behaviour and leak detection in pipelines. Computers & chemical engineering 20, S491–S496.

Srinivasan, R., Rengaswamy, R., 2005. Stiction compensation in process control loops: A framework for integrating stiction measure and compensation. Industrial and Engineering Chemistry Research 44 (24), 9164–9174.

Srinivasan, R., Rengaswamy, R., 2006. Integrating stiction diagnosis and stiction compensation in process control valves. Computer Aided Chemical Engineering 21, 1233–1238.

Srinivasan, R., Rengaswamy, R., 2008. Approaches for efficient stiction compensation in process control valves. Computers and Chemical Engineering 32 (1), 218–229.

Thomas, P. J., 1999. Simulation of industrial processes for control engineers. Butterworth-Heinemann, Chater 19, 239-243.

Uilhoorn, F. E., 2014. State-space estimation with a bayesian filter in a coupled pde system for transient gas flows. Applied Mathematical Modelling.

Uilhoorn, F. E., 2015. State-space estimation with a bayesian filter in a coupled pde system for transient gas flows. Applied Mathematical Modelling 39 (2), 682–692.

van den Boom, T. J., Backx, T., 2010. Model predictive control. DISC Course, Lecture Notes 16.

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., 2003a. A review of process fault detection and diagnosis part ii: Qualitative models and search strategies. Computers and Chemical Engineering 27 (3), 313–326.

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., Yin, K., 2003b. A review of process fault detection and diagnosis: Part iii: Process history based methods. Computers & chemical engineering 27 (3), 327–346.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., Kavuri, S. N., 2003c. A review of process fault detection and diagnosis: Part i: Quantitative model-based methods. Computers & chemical engineering 27 (3), 293–311.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., Kavuri, S. N., 2003d. A review of process fault detection and diagnosis: Part i: Quantitative model-based methods. Computers & chemical engineering 27 (3), 293–311.

Verde, C., 2001. Multi-leak detection and isolation in fluid pipelines. Control Engineering Practice 9 (6), 673–682.

Vershik, A. M., 2006. Kantorovich metric: Initial history and little-known applications. Journal of Mathematical Sciences 133 (4), 1410–1417.

Wald, A., 1973. Sequential analysis. Courier Corporation.

Watanabe, K., Koyama, H., Tanoguchi, H., Ohma, T., Himmelblau, D., 1993. Location of pinholes in a pipeline. Computers & chemical engineering 17 (1), 61–70.

Willis, A., 2011. Design of a modified sequential probability ratio test (sprt) for pipeline leak detection. Computers & chemical engineering 35 (1), 127–131.

Wise, B. M., Gallagher, N. B., 1996. The process chemometrics approach to process monitoring and fault detection. Journal of Process Control 6 (6), 329–348.

Xiang Ivan, L. Z., Lakshminarayanan, S., 2009. A new unified approach to valve stiction quantification and compensation. Industrial and Engineering Chemistry Research 48 (7), 3474–3483.

Xie, L., C. Y. H. A., 2013. An improved valve stiction simulation model based on isa standard tests. Control Engineering Practice 21, 1359–1368.

Zhang, J., 1997. Designing a cost-effective and reliable pipeline leak-detection system. Pipes and Pipelines International 42 (1), 20–26.

Zhang, J., Lv, C., Yue, X., Li, Y., Yuan, Y., 2014. Study on a linear relationship between limited pressure difference and coil current of on/off valve and its influential factors. ISA transactions 53 (1), 150–161.

Zhang, T., Tan, Y., Zhang, X., Zhao, J., 2015. A novel hybrid technique for leak detection and location in straight pipelines. Journal of Loss Prevention in the Process Industries 35, 157–168.

Zhu, J., Shu, Y., Zhao, J., Yang, F., 2014. A dynamic alarm management strategy for chemical process transitions. Journal of Loss Prevention in the Process industries 30, 207–218.

# Appendices

# Appendix A

# More Results of Change Detection Algorithm on Tennessee Eastman Process

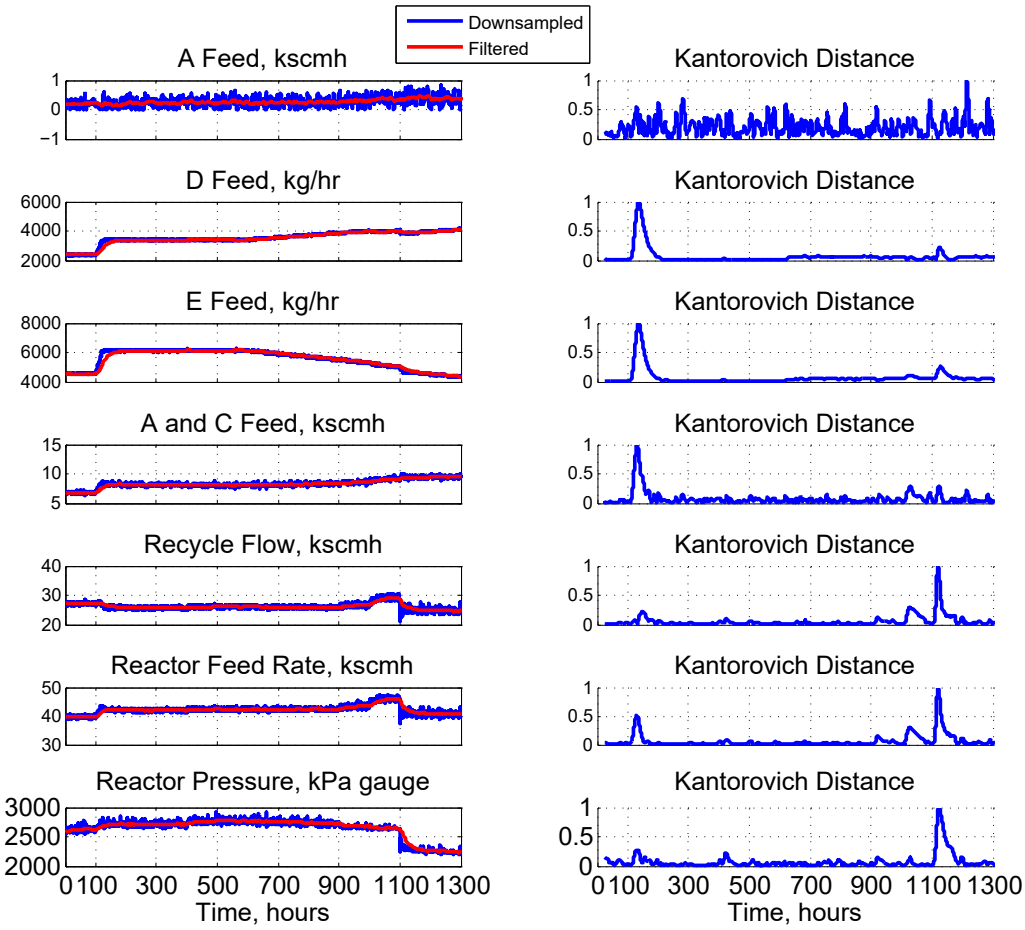


**Figure A.1:** Measurements 1 to 7 from TE process and their corresponding normalized KD scores.

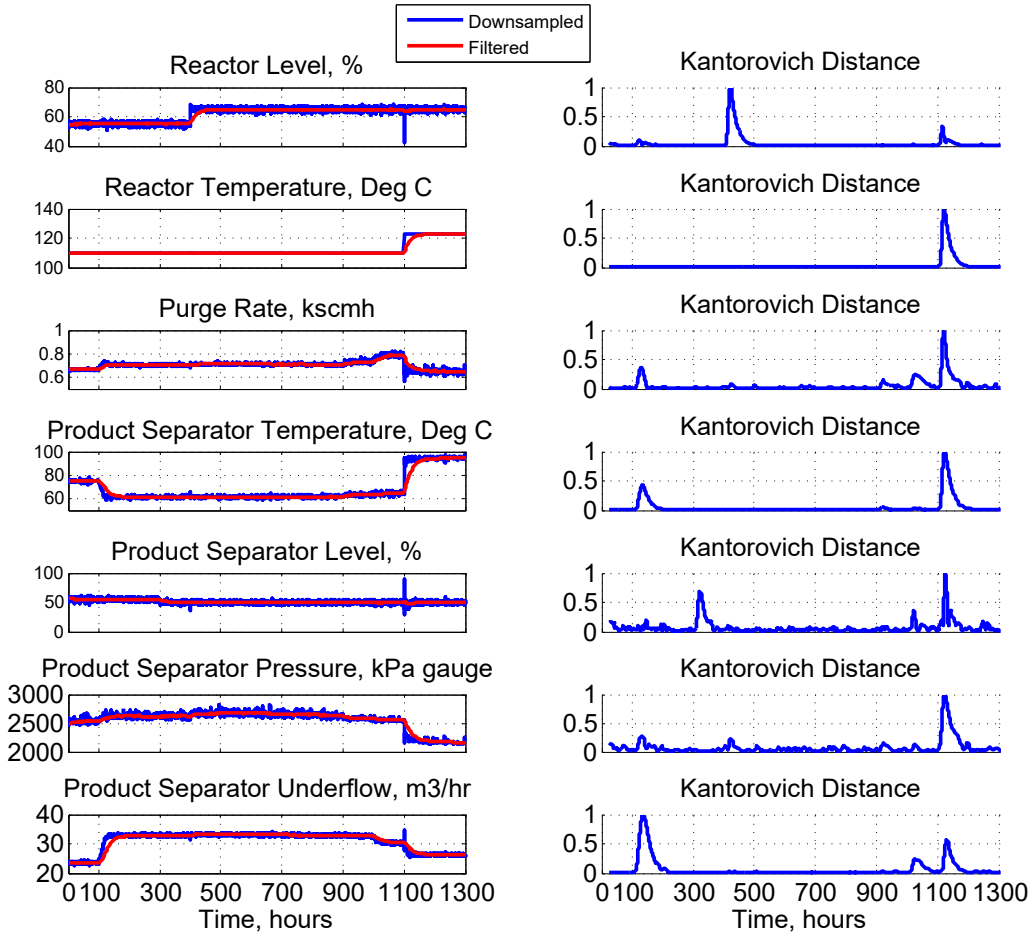


**Figure A.2:** Measurements 8 to 14 from TE process and their corresponding normalized KD scores.

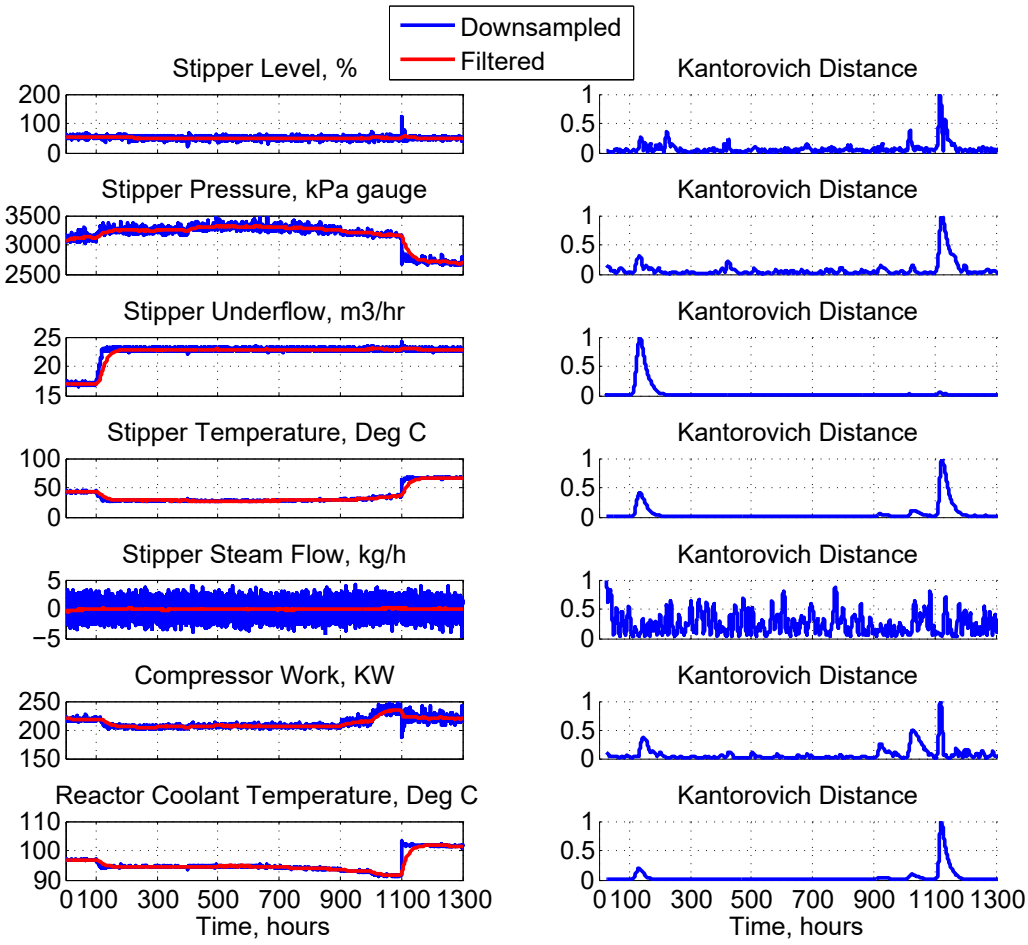


**Figure A.3:** Measurements 15 to 21 from TE process and their corresponding normalized KD scores.

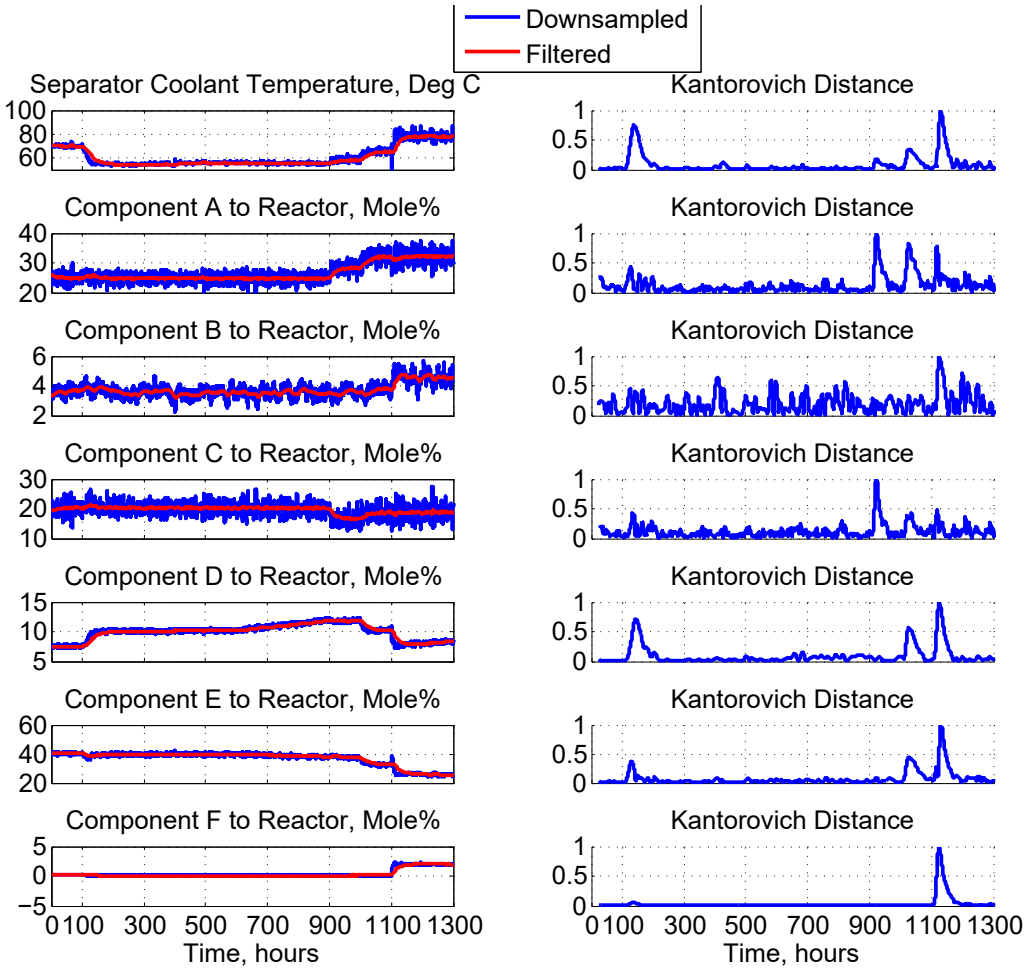


**Figure A.4:** Measurements 22 to 28 from TE process and their corresponding normalized KD scores.

# Appendix B

# Development of Pipeline Models

A long pipeline is inherently distributed in nature, that is fluid properties vary with the pipe length and it is not possible to get bulk fluid properties. In this chapter, mathematical derivation of the conservation of mass, momentum and energy equations for a pipeline are discussed in details.

## B.1 Conservation of Mass in a Pipeline

The principle of conservation of mass can be stated as follows,
*Mass accumulation = mass in flow-rate − mass out flow-rate*
Mathematically, the conservation of mass principle can be expressed by equation B.1.

$$\frac{\partial m}{\partial t} = \sum W_{in} - \sum W_{out} \tag{B.1}$$

Following assumptions were made while developing the conservation of mass equation -

- The fluid is homogenous that is flow is of uniform velocity across the pipe.
- Flow is parallel to the axis of pipe and flow is one-dimensional.
- All variables are functions of both time and distance.
- Pipe cross sectional area is constant throughout the length
- Temperature is constant throughout the length

2) Flow is parallel to the axis of pipe and flow is one-dimensional.
3) All variables are functions of both time and distance.
4) Pipe cross sectional area is constant throughout the length
5) Temperature is constant throughout the length

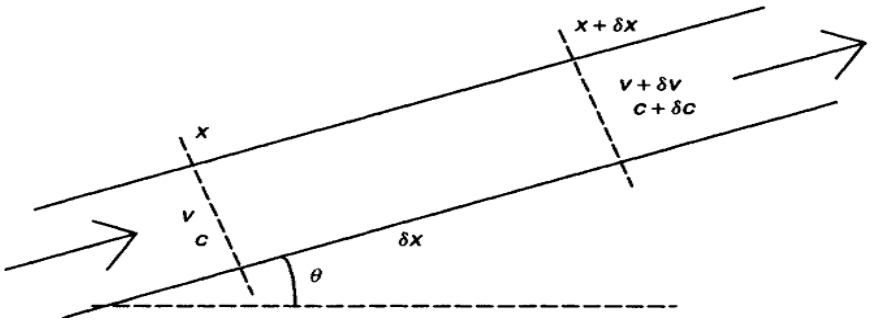


*Figure 1: Element of pipe at an angle θ to derive the continuity equation (Courtesy : Philip L. Thomas, 1999).*

**Figure B.1:** Element of pipe at an angle θ to derive the continuity equation (Courtesy : Thomas, 1999). *Simulation of Industrial Processes for Control Engineers, Elsevier, 1999)*

At time $t$ and reference position ,

fluid specific volume $= v\ (\frac{m^3}{kg})$

fluid velocity $= c\ (\frac{m}{s})$

At time $t$ and reference position , fluid specific volume = $v(\frac{m^3}{kg})$ fluid velocity = $c(\frac{m}{s})$.

At the same time instant at reference position $x + \delta x$, we have

specific volume $= v + \delta v = v + \frac{\partial v}{\partial x}\delta x$ ……(1)

velocity $= c + \delta c = c + \frac{\partial c}{\partial x}\delta x$

The mass flow into the pipe element can be shown by equation B.2.

The mass flow into the pipe element is $W_{in} = A\frac{c}{v}$ ……(3)

$$W_{in} = A\frac{c}{v} \tag{B.2}$$

where $A$ is the pipe cross-sectional area.

Similarly mass out of the pipe element can be expressed by equation B.3.

Similarly mass out of the pipe element is $W_{out} = A\frac{c}{v} + \frac{\partial}{\partial x}(A\frac{c}{v})\,\delta x$ ……(4)

$$W_{out} = A\frac{c}{v} + \frac{\partial}{\partial x}(A\frac{c}{v})\delta x \tag{B.3}$$

Fluid mass in the pipe element $m = A\frac{\delta x}{v}$

Fluid mass in the pipe element can be expressed by $m = A\frac{\delta x}{v}$. So, mass accumulation in the pipe element can be expressed by equation B.4.

So, $\frac{dm}{dt} = \frac{d}{dt}(A\frac{\delta x}{v})$ ……(5)

Now by applying principle of conservation of mass we have ,

Mass accumulation = mass in flow rate – mass out flow rate

$$\frac{dm}{dt} = \frac{d}{dt}(A\frac{\delta x}{v}) \tag{B.4}$$

Now, combining equations B.2, B.3 and B.4 with equation B.1, the conservation of mass equation for the pipe element can be reformulated as equation B.5.

$$\frac{d}{dt}(A\frac{\delta x}{v}) = -\frac{\partial}{\partial x}(A\frac{c}{v})\delta x \tag{B.5}$$

Since, $\delta x$ does not vary with time equation B.5 can be further simplified as in equation B.6

$$\frac{\partial}{\partial t}(\frac{A}{v}) = -\frac{\partial}{\partial x}(A\frac{c}{v}) \tag{B.6}$$

Since, the pipeline cross-sectional area is also independent of time this equation can be even more simplified as in equation B.7.

$$\frac{\partial \nu}{\partial t} = \nu^2 \frac{\partial}{\partial x}(\frac{c}{\nu}) \tag{B.7}$$

With the assumption of constant temperature, the following two conclusions can be made -

- $\nu = \nu(p)$
- $p = p(x, t)$

Now,
$\frac{\partial \nu}{\partial t} = \frac{d\nu}{dp}\frac{\partial p}{\partial t}$.
Speed of sound in the fluid can be defined as $C_{son} = \sqrt{\nu\kappa}$.
Again, the bulk modulus of elasticity of fluid is defined as $\kappa = -\nu\frac{dp}{d\nu}$.
Hence, $\frac{d\nu}{dp} = -\frac{\nu}{\kappa} = -\frac{\nu^2}{\nu\kappa} = -\frac{\nu^2}{C_{son}^2}$.

Combining these results with equation B.7, the following equation can be derived -

$$\begin{aligned} \frac{\partial \nu}{\partial t} &= \frac{d\nu}{dp}\frac{\partial p}{\partial t} \\ \nu^2 \frac{\partial}{\partial x}(\frac{c}{\nu}) &= -\frac{\nu^2}{C_{son}^2}\frac{\partial p}{\partial t} \\ \frac{\partial p}{\partial t} + C_{son}^2 \frac{\partial}{\partial x}(\frac{c}{\nu}) &= 0 \end{aligned} \tag{B.8}$$

Again,

$$\begin{aligned} W &= \frac{Ac}{\nu} \\ \Rightarrow \frac{\partial W}{\partial x} &= A\frac{\partial}{\partial x}(\frac{c}{\nu}) \end{aligned} \tag{B.9}$$

So, equation B.8 can be reformed as following -

$$\frac{\partial p}{\partial t} + \frac{C_{son}^2}{A}\frac{\partial W}{\partial x} = 0 \tag{B.10}$$

Equations B.8 and B.10 are two forms of continuity equation for long pipeline.

## B.2 Conservation of Momentum in a Pipeline

Conservation of momentum for the pipe element of length $\delta x$ can be expressed by the following statement :

*Change in momentum = momentum brought in by incoming fluid − momentum leaving with outgoing fluid + impulse of force due to pressure on the fluid at inlet − impulse of force due to pressure on the fluid at outlet − impulse of the frictional force that acts against the flow − impulse of the gravitational force acting against flow .*

Conservation of momentum can be expressed mathematically as in equation B.11.

$$\delta\left(\frac{A\delta x}{v}c\right) = A\delta x\frac{\partial}{\partial x}\left(\frac{Ac}{v}\right)\delta tc - \left[\frac{Ac}{v}\delta tc + \frac{\partial}{\partial x}\left(\frac{Ac}{v}\delta tc\right)\delta x\right] + pA\delta t - \left[pA\delta t + \frac{\partial}{\partial x}(pA\delta t)\delta x\right] - \tau\pi D\delta x\delta t - \frac{A\delta x}{v}g\sin\theta\delta t \quad \text{(B.11)}$$

where,

- $D$ = internal diameter of the pipe ($m$),
- $\tau = \frac{1}{2}\frac{c|c|}{v}f$ = frictional shear stress ($N/m^2$),
- $f$ = dimensionless Fanning friction factor

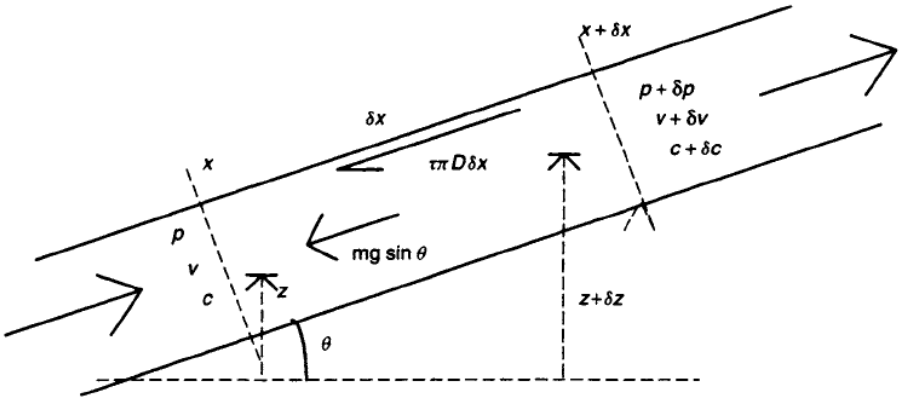


*Figure 12: Forces acting on the element of pipe at an angle $\theta$ to derive the momentum equation (Courtesy : Philip J. Thomas, Simulation of Industrial Processes for Control Engineers, Elsevier, 1999)*

**Figure B.2:** Forces acting on the element of pipe at an angle to derive the momentum equation (Courtesy : Thomas, 1999).

Using the definition of $\tau$ in equation B.11 the equation becomes as follows after some modifications

$$\frac{\partial}{\partial t}\left(\frac{W}{A}\right) + \frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\left(\frac{W^2}{A^2}v\right) + \frac{2vW|W|}{A^2D}f + \frac{g}{v}\sin\theta = 0 \quad \text{.....(24)}$$

Now, $\frac{\partial}{\partial x}\left(\frac{W^2}{A^2}v\right) = \frac{W^2}{A^2}\frac{\partial v}{\partial x} + 2v\frac{W}{A^2}\frac{\partial W}{\partial x}$ ....(25)

$$[\because \frac{dv}{dp} = -\frac{v^2}{c_{son}{}^2}]$$



Combining equations 24 and 25 we have

$$\frac{1}{A}\frac{\partial W}{\partial t} + \frac{\partial p}{\partial x}\left(1 - \frac{v^2}{c_{son}{}^2}\frac{W^2}{A^2}\right) + 2v\frac{W}{A^2}\frac{\partial W}{\partial x} + \frac{2vW|W|}{A^2D}f + \frac{g}{v}\sin\theta = 0 \quad \text{.....(26)}$$

Now dividing both sides of equation B.12 by $A\delta x\delta t$ and then taking $\delta t \to 0$ with $A = \frac{\pi D^2}{4}$, the following simplified equation is found. Equation B.13 is one of forms of momentum conservation equation which is frequently used in literature.

$$\frac{\partial}{\partial t}(\frac{c}{\nu}) = -\frac{\partial}{\partial x}(\frac{c^2}{\nu}) - \frac{\partial}{\partial x}(p) - \frac{1}{2}\frac{c|c|\pi D}{A\nu}f - \frac{g}{\nu}sin\theta$$
$$\frac{\partial}{\partial t}(\frac{c}{\nu}) = -\frac{\partial}{\partial x}(\frac{c^2}{\nu} + p) - \frac{2c|c|}{\nu D}f - \frac{g}{\nu}sin\theta \tag{B.13}$$

To derive the other form of momentum equation, $W = \frac{Ac}{\nu}$ is used in equation B.13.

$$\frac{\partial}{\partial t}(\frac{W}{A}) + \frac{\partial p}{\partial x} + \frac{\partial}{\partial x}(\frac{W^2}{A^2}\nu) + \frac{2\nu W|W|}{A^2 D}f + \frac{g}{\nu}sin\theta = 0 \tag{B.14}$$

where,

$$\frac{\partial}{\partial x}(\frac{W^2}{A^2}\nu) = \frac{W^2}{A^2}\frac{\partial \nu}{\partial x} + 2\nu\frac{W}{A^2}\frac{\partial W}{\partial x}$$
$$\Rightarrow \frac{\partial}{\partial x}(\frac{W^2}{A^2}\nu) = \frac{W^2}{A^2}\frac{d\nu}{dp}\frac{\partial p}{\partial x} + 2\nu\frac{W}{A^2}\frac{\partial W}{\partial x}$$

Again,

$$\frac{d\nu}{dp} = -\frac{\nu^2}{C_{son}^2}$$
$$\therefore \frac{\partial}{\partial x}(\frac{W^2}{A^2}\nu) = -\frac{\nu^2}{C_{son}^2}\frac{W^2}{A^2}\frac{\partial p}{\partial x} + 2\nu\frac{W}{A^2}\frac{\partial W}{\partial x} \tag{B.15}$$

Combining equations B.14 and B.15, second form of the momentum equation is obtained as in equation B.16.

$$\frac{1}{A}\frac{\partial W}{\partial t} + \frac{\partial p}{\partial x}(1 - \frac{\nu^2}{C_{son}^2}\frac{W^2}{A^2}) + 2\nu\frac{W}{A^2}\frac{\partial W}{\partial x} + \frac{2\nu W|W|}{A^2 D}f + \frac{g}{\nu}sin\theta = 0 \tag{B.16}$$

# Appendix C

# Methods for Handling the Degeneracy Problem of SIS Particle Filter

## C.1 Resampling Algorithm

The basic idea of rasampling algorithm is that whenever a significant degeneracy is observed then the particles with negligible weights are discarded and the particles with large weight will be given more emphasis. A new set of particles ($\{x_k^{i*}\}_{j=1}^{N_S}$) is generated by resampling (with replacement) $N_S$ times from a discrete approximation of $p(x_k|z_{1:k})$ as given in equation C.1 (Arulampalam et al., 2002).

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_S} w_k^i \delta(x_k - x_k^i) \tag{C.1}$$

This equation infers that $Pr(x_k^{i*} = x_k^j) = w_k^j$. The samples generated in this are i.i.d samples from the discrete density of equation C.1. The weights are reset to $w_k^i = \frac{1}{N_S}$. This resampling procedure can be implemented by $O(N_S)$ mathematical operations by sampling $N_S$ ordered uniforms (Arulampalam et al., 2002).

### C.1.1 Generic Particle Filter

The generic particle filter uses the resampling stage to overcome the degeneracy problem. The pseudocode of a generic particle filter is given in algorithm 4 (Ristic et al., 2004). Clearly the prediction ($p(x_k|z_{k-1})$) step is done as the $SIS$ algorithm (algorithm 1). Then with the measurement ($z_k$) available at hand, the importance weight of each of the particle is calculated. Thus an estimate of posterior distribution ($p(x_k|z_k)$) is obtained by approximating the likelihood ($p(z_k|x_{k-1})$).

If $\widehat{N_{eff}} < N_S$, the resampling step is performed according to algorithm 3 to improve the

**Algorithm 3** : Resampling Algorithm

$[\{x_k^{j*}, w_k^j, i^j\}_{j=1}^{N_S}] = RESAMPLE[\{x_k^i, w_k^i\}_{i=1}^{N_S}]$
Initialize the CDF: $c_1 = 0$
**for** $i = 2 : N_S$ **do**
    Construct CDF: $c_i = c_{i-1} + w_k^i$
**end for**
Start at the bottom of the CDF: $i = 1$
Draw a startign point: $u_1 \sim \mathbb{U}[0, N_S^{-1}]$
**for** $j = 1 : N_S$ **do**
    Move alogn the CDF: $u_j = u_1 + N_S^{-1}(j-1)$
    **while** $u_j > c_j$ **do**
        $i = i + 1$
    **end while**
    Assign sample: $x_k^{j*} = x_k^i$
    Assign weight: $w_k^j = N_S^{-1}$
    Assign parent: $i^j = i$
**end for**

degeneracy. So, important particles are kept and particles with less weight are discarded. Though resampling algorithm reduces the effect of degeneracy, it has some demerits too.

**Algorithm 4** : Generic Particle Filter

$[\{x_k^i, w_k^i\}_{i=1}^{N_S}] = PF[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_S}, z_k]$
Filtering via SIS (Algorithm 1): $[\{x_k^i, w_k^i\}_{i=1}^{N_S}] = PF[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_S}, z_k]$
Calculate $\widehat{N_{eff}}$ using equation 4.22
**if** $\widehat{N_{eff}} < N_S$ **then**
    Resample using algorithm 3

Firstly, since all the particles must be combined, the opportunity to parallelize the implementation becomes limited. Secondly, since only the highest weighted particles are kept, diversity of particles is lost. For the same reason, sometimes the resultant sample may contain repeated points. This phenomena is knows as 'sample impoverishment'. This phenomena is pronounced heavily when the magnitude of process noise is very small (Arulampalam et al., 2002; Ristic et al., 2004). In such cases, number of particles will reduce to a single particle within few iterations. Thirdly, due to diversity reduction of paths of particles, smoothed estimates based on particles' paths degenerate (Ristic et al., 2004).

One way to tackle these issues is using the Markov chain Monte Carlo (MCMC) move step (Ristic et al., 2004). Another class of particle filter that can handle this issue is the regularized particle filter (Arulampalam et al., 2002; Ristic et al., 2004).

## C.2 Good Choice of Importance Density

Another way to reduce the effect of degeneracy is a good choice of importance density. Choosing a good importance density is one of the most critical issues in designing a particle filter. The basic idea is to choose the importance density ($q(x_k|x^i_{k-1}, z_k)$) in such way so that Var$w^{*i}_k$ is minimized and hence, $N_{eff}$ is maximized. The optimal importance density can expressed by equation C.2 (Arulampalam et al., 2002; Ristic et al., 2004).

$$\begin{aligned} q(x_k|x^i_{k-1}, z_k)_{opt} &= p(x_k|x^i_{k-1}, z_k) \\ q(x_k|x^i_{k-1}, z_k)_{opt} &= \frac{p(z_k|x_k, x^i_{k-1})p(x_k|x^i_{k-1})}{p(z_k|x^i_{k-1})} \end{aligned} \tag{C.2}$$

Substitution of equation C.2 into equation 4.18 gives

$$\begin{aligned} w^i_k &\infty w^i_{k-1} p(z_k|x^i_{k-1}) \\ w^i_k &= w^i_{k-1} \int p(z_k|x_k)p(x_k|x^i_{k-1})dx_k \end{aligned} \tag{C.3}$$

The importance density referred by equation C.3 is an optimal choice because for a given $x^i_{k-1}$, $w^i_k$ takes the same value regardless the sample drawn from $q(x_k|x^i_{k-1}, z_k)_{opt}$. Hence, Var$w^{*i}_k$ becomes 0 for the given $x^i_{k-1}$. So, this is only conditional to $x^i_{k-1}$.

The optimal importance density has two limitations.

- Sampling must be available from $p(x_k|x^i_{k-1}, z_k)$
- The integration should be evaluated over the new state.

In general, either of these two may not be feasible. However, the optimal importance density can be applied in the two cases -

- When $x_k$ is a member of a finite set, the integration becomes a sum and sampling from $p(x_k|x^i_{k-1}, z_k)$ is possible.
- If $p(x_k|x^i_{k-1}, z_k)$ is Gaussian, analytical evaluation is possible.

# Appendix D

# More Case Studies on Model Based Leak Detection Algorithm

## D.1 Results for Leak at Node 6

Figures D.1 to D.4 describes the proposed model-based LDS algorithm (section 4.2) for 5% leak at node 6 at 450s. Figure D.1 and D.2 gives the residuals between simulated pipeline and PF estimation of the pressure and flow-rates at nodes 1, 5, 10, 15 and 21. Figure D.3 gives the corresponding KD calculation for the pressure measurements. Finally, Figure D.4 gives the localization results which indicates that leak is in between node 5 and 10 and nearer to node 5.

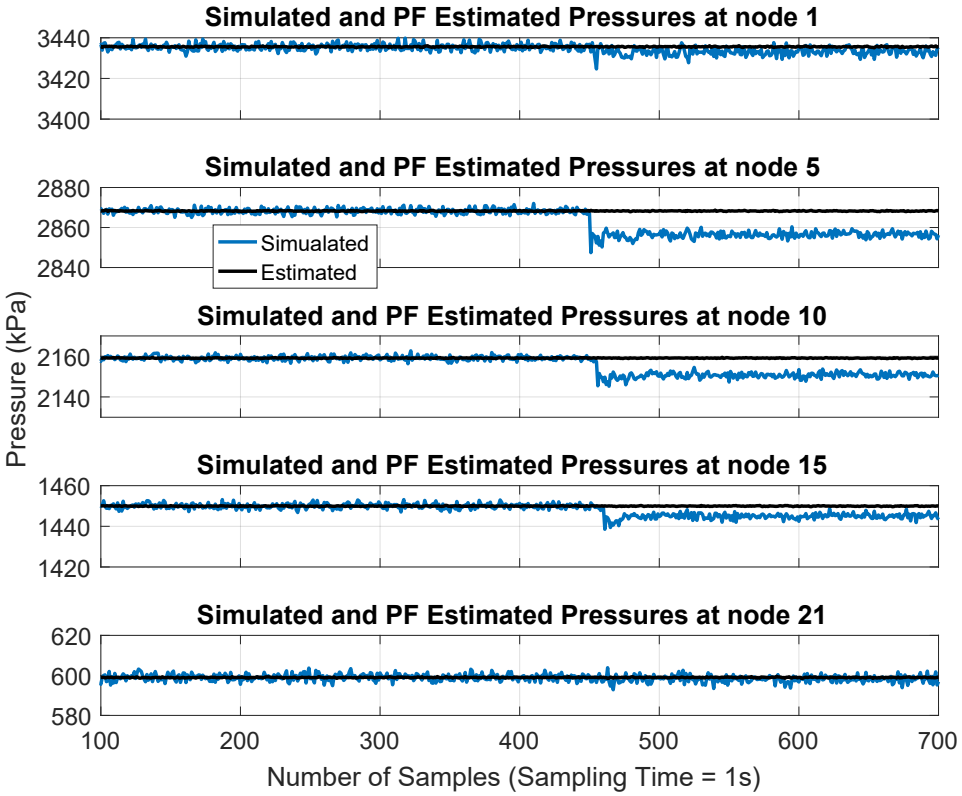


**Figure D.1:** Residuals between the PF estimated pressures and available true (simulated) pressure measurements for 5% leak at node 6 at 450 s.

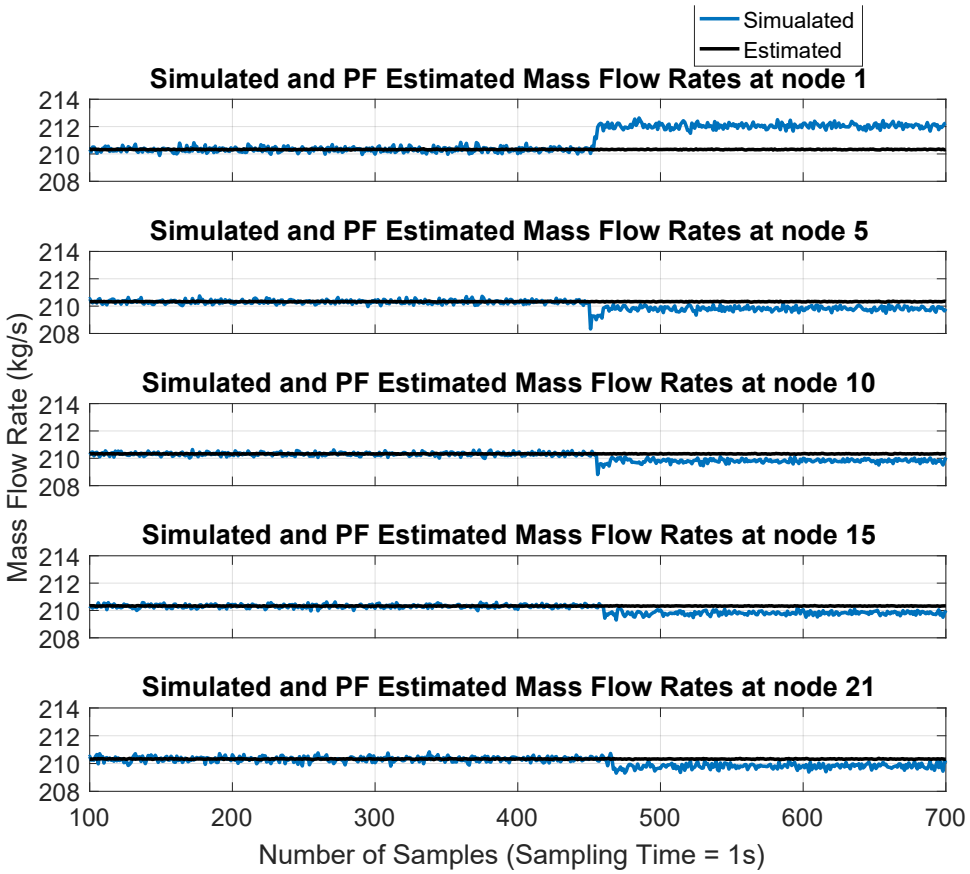


**Figure D.2:** Residuals between the PF estimated mass flow-rates and available true (simulated) flow-rates for 5% leak at node 6 at 450 s.

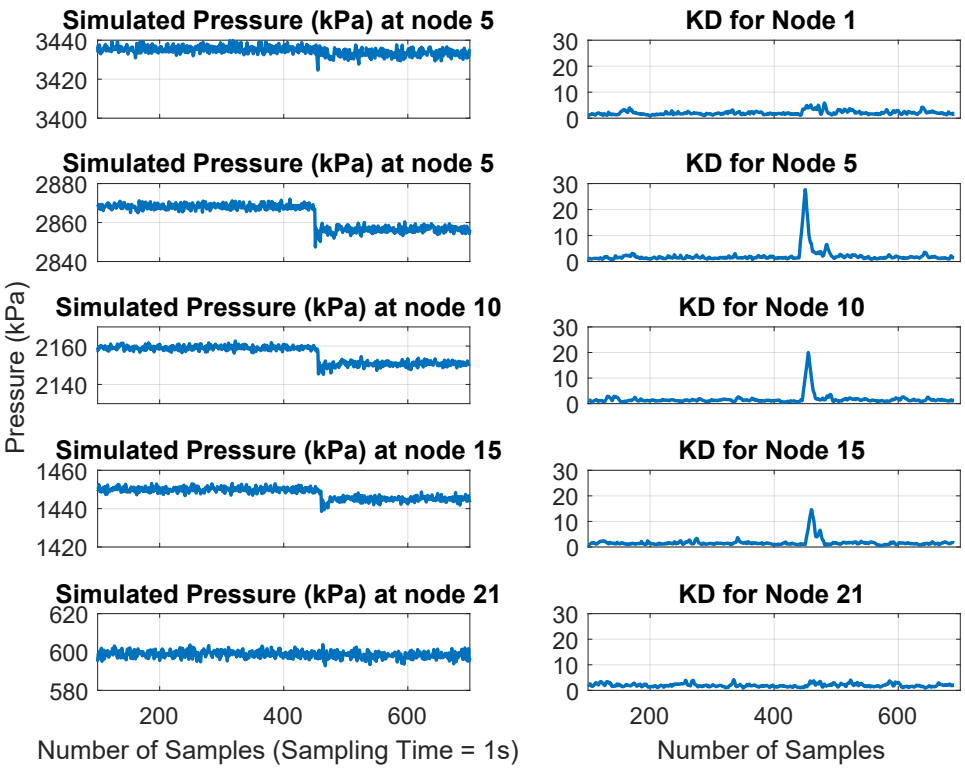


**Figure D.3:** Detecting change time point using Kantorovich distance algorithm on the available true (simulated) pressure measurements for 5% leak at node 6 at 450 s.

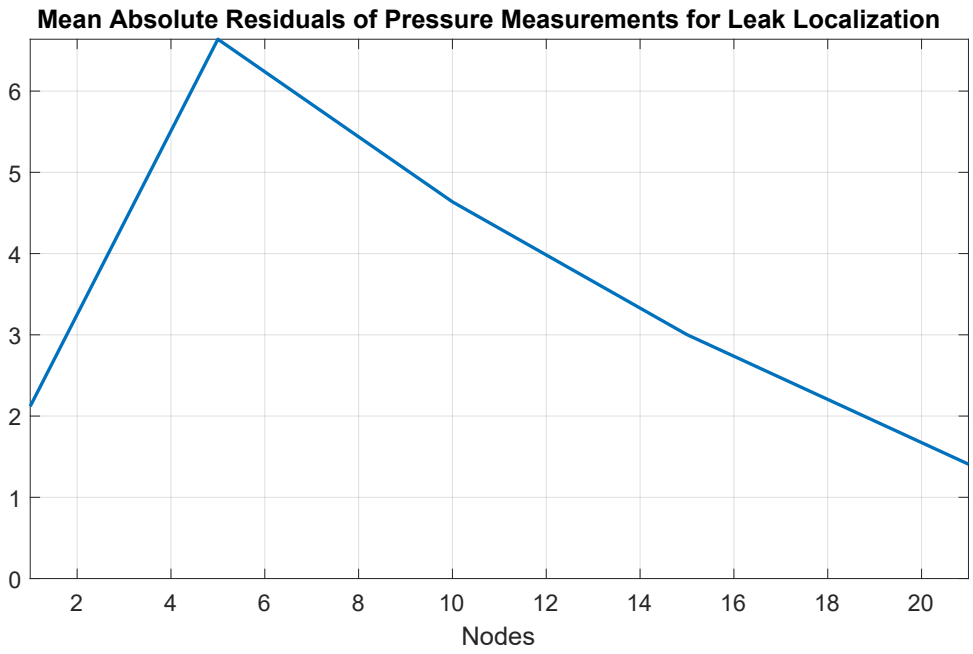


**Figure D.4:** Leak localization using the mean absolute residual calculated around detected change time point for 5% leak at node 6 at 450 s.

## D.2 Results for Leak at Node 9

This section describes the detection and localization results of 3% leak at node 9 at 450s using the proposed model-based LDS algorithm (section 4.2). Figure D.5 and D.6 gives the residuals between simulated pipeline (simulated) and PF estimation (estimated) of the pressure and flow-rates at nodes 1, 5, 10, 15 and 21. Figure D.7 gives the corresponding KD calculation for the pressure measurements.

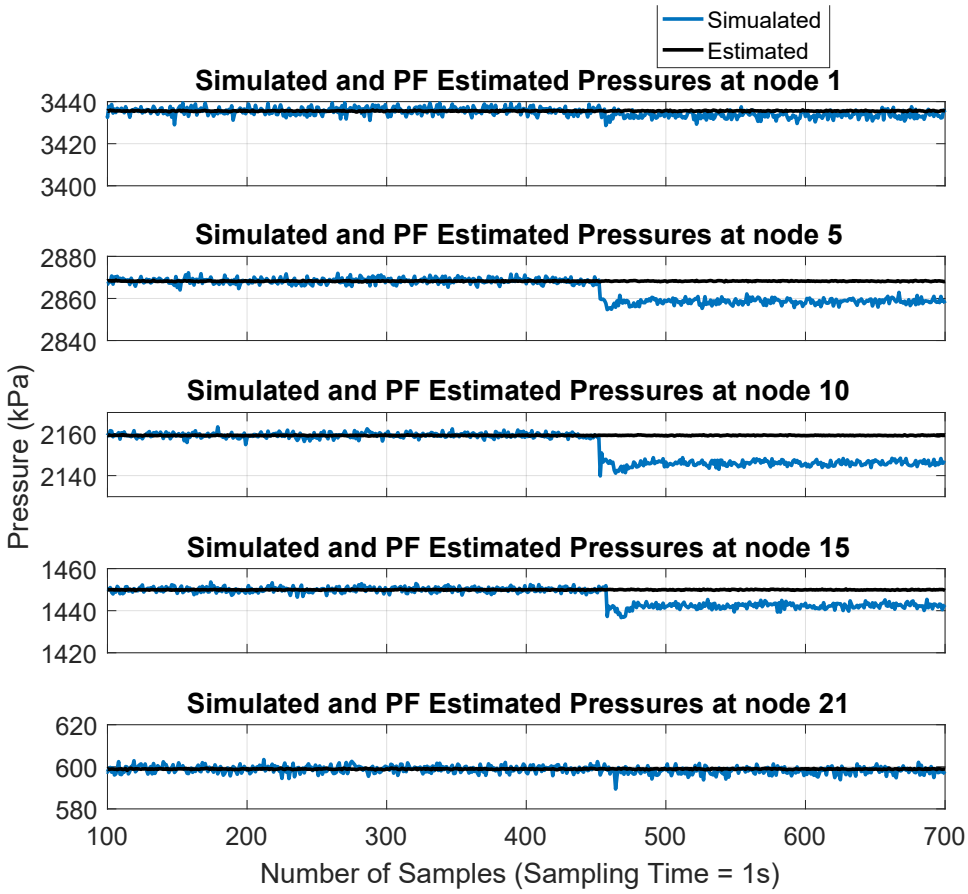


**Figure D.5:** Residuals between the PF estimated pressures and available true (simulated) pressure measurements for 3% leak at node 9 at 450 s.

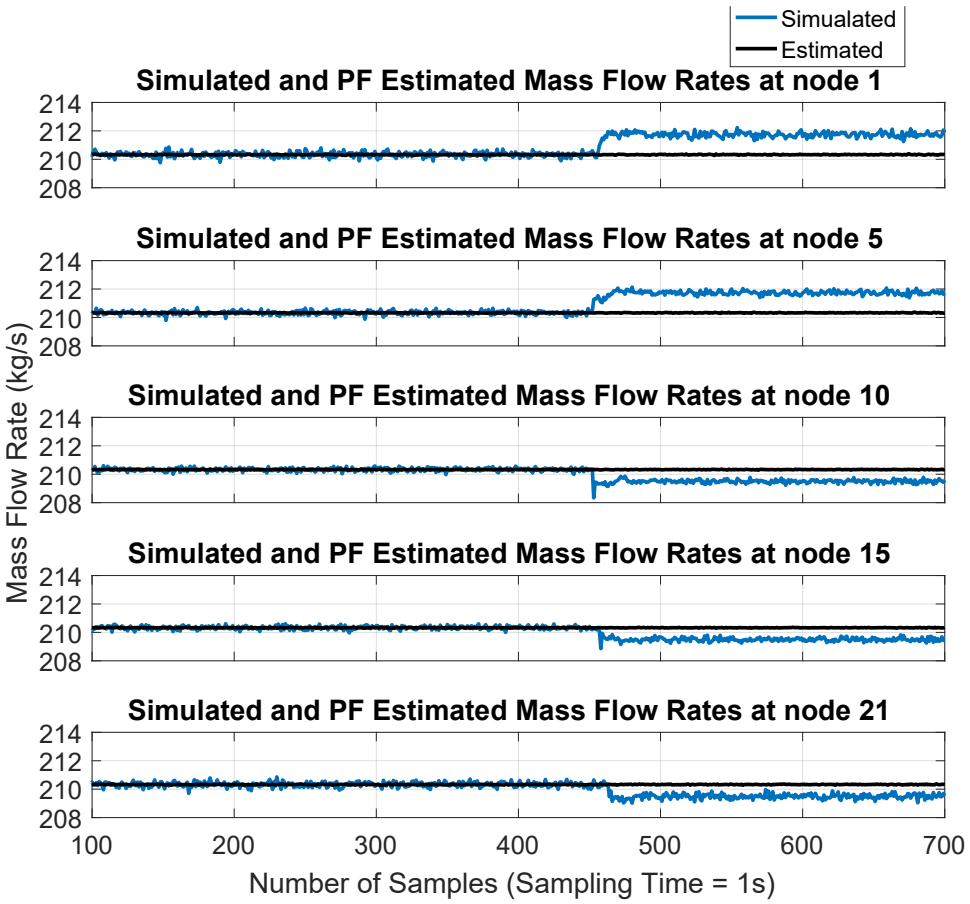


**Figure D.6:** Residuals between the PF estimated mass flow-rates and available true (simulated) flow-rates for 3% leak at node 9 at 450 s.

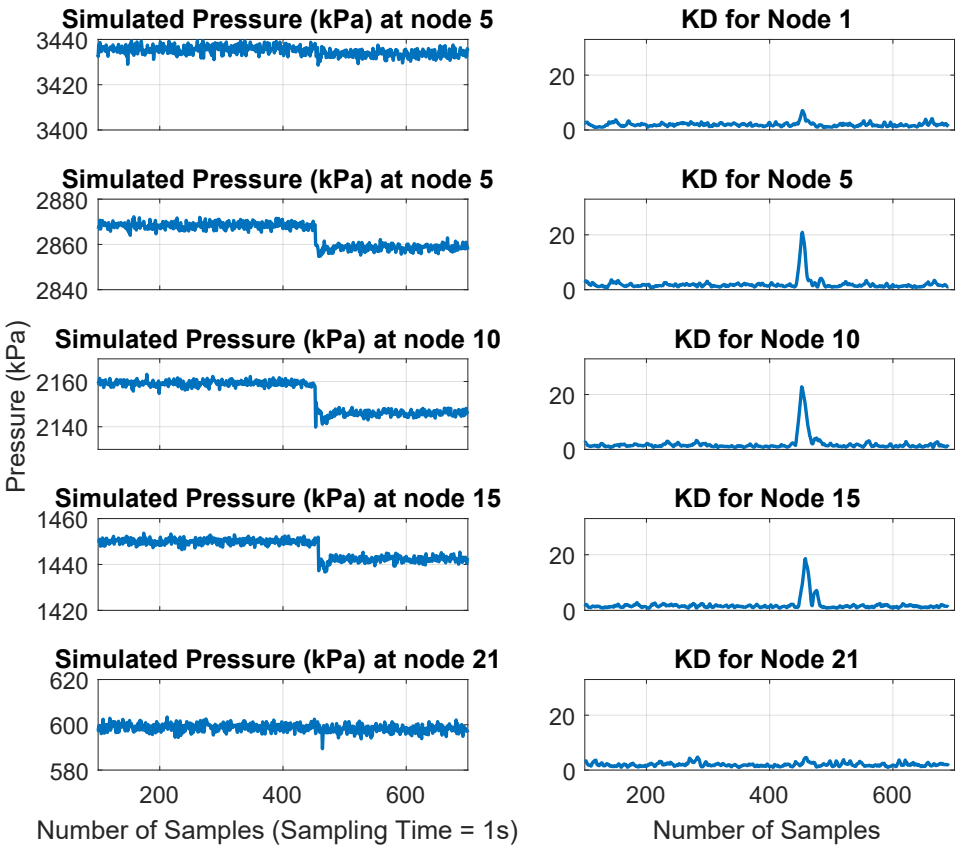


**Figure D.7:** Detecting change time point using Kantorovich distance algorithm on the available true (simulated) pressure measurements for 3% leak at node 9 at 450 s.

Figure D.8 gives the localization results which indicates that leak is in between node 5 and 10 and nearer to node 10.

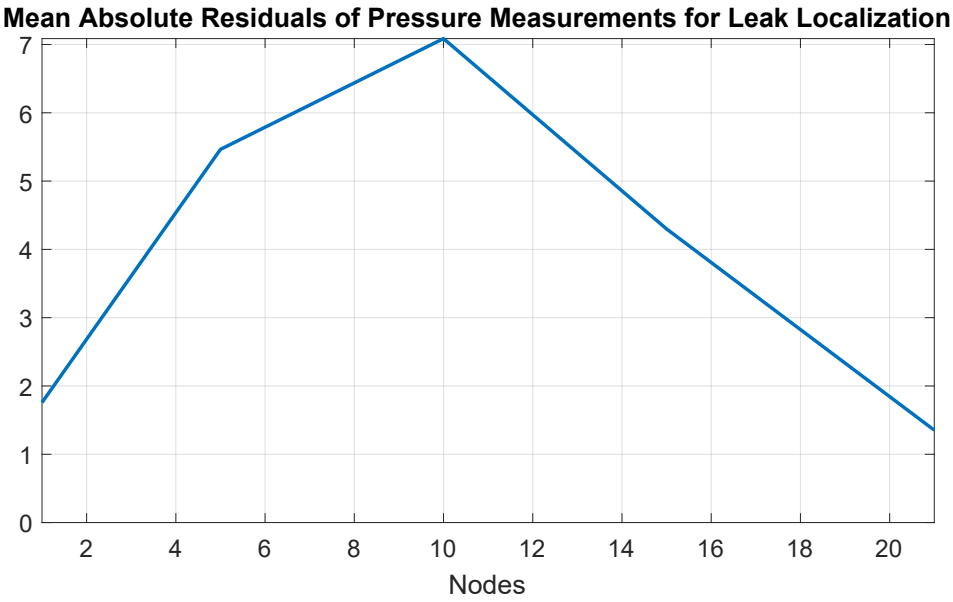


**Figure D.8:** Leak localization using the mean absolute residual calculated around detected change time point for 3% leak at node 9 at 450 s.