8-2004

Inferential Modeling and Independent Component Analysis for Redundant Sensor Validation

Jun Ding
University of Tennessee - Knoxville

To the Graduate Council:

I am submitting herewith a dissertation written by Jun Ding entitled "Inferential Modeling and Independent Component Analysis for Redundant Sensor Validation." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Nuclear Engineering.

J. Wesley Hines, Major Professor

We have read this dissertation and recommend its acceptance:

Robert E. Uhrig, Belle R. Upadhyaya, Charles F. Moore

Accepted for the Council:
Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a dissertation written by Jun Ding entitled "Inferential Modeling and Independent Component Analysis for Redundant Sensor Validation." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirement for the degree of Doctor of Philosophy, with a major in Nuclear Engineering.

J. Wesley Hines
Major Professor

We have read this dissertation
and recommend its acceptance:

Robert E. Uhrig

Belle R. Upadhyaya

Charles F. Moore

Accepted for the Council:

Anne Mayhew
Vice Chancellor and Dean of
Graduate Studies

(Original Signatures are on file with official student records.)

INFERENTIAL MODELING AND INDEPENDENT COMPONENT ANALYSIS FOR REDUNDANT SENSOR VALIDATION

A Dissertation
Presented for the
Doctor of Philosophy Degree

The University of Tennessee, Knoxville

Jun Ding
August 2004

DEDICATION

for Jing

ACKNOWLEDGMENTS

I would like to thank the members of my dissertation committee. I wish to thank Dr. J. Wesley Hines, my major professor, for his guidance, patience and encouragement. I sincerely appreciate that he created the opportunity for me to begin a Ph.D program in nuclear engineering. I remember thousands miles free ride to conferences; Dr. Hines was always behind the wheel. I am indebted to him so much for making the journey of study and research interesting and fruitful. I thank Dr. Robert E. Uhrig for sharing his insights, vision, and especially his scientific attitude towards research. I enjoyed weekly group meeting, where ideas spark and results have been criticized. Dr. Belle R. Upadhyaya, who has taken the time to review this work and has always been ready and willing to help out along the way. Dr. Charles F. Moore, who has brought his experience in practical application to the table in reviewing and guiding this work.

I am grateful to all the other wonderful faculty members in nuclear engineering department in University of Tennessee. I thank Dr. Gribok for his supervision of my early work. I would like to thank Dr. Ronald E. Pevey, Dr. Laurence F. Miller, Dr. Lawrence W. Towsend, Dr. Peter G. Groer and Dr. Arthur E. Ruggles, for their teaching and discussions. I would like to thank Dr. Dodds, the department head, for creating a wonderful academic environment in the department.

I am thankful to Dr. Brandon Rasmussen, for his help and discussions.

I would like to acknowledge the Electric Power Research Institute for funding early work in this area and also to acknowledge the Tennessee Valley Authority for their follow-on support.

I am thankful to my classmates, especially to Xuedong Huang, Bofu Lu and Ke Zhao. I would like thank Nathan DeLauder, for his help and suggestions.

At last but not least I would like to thank my lovely wife Jing for her encouragement, understanding and patience. I would like to thank my seven year old son Chunyang, for being with me through some of my Ph.D classes during school in-service days, for his patience and not getting bored. Thank you all.

ABSTRACT

The calibration of redundant safety critical sensors in nuclear power plants is a manual task that consumes valuable time and resources. Automated, data-driven techniques, to monitor the calibration of redundant sensors have been developed over the last two decades, but have not been fully implemented. Parity space methods such as the Instrumentation and Calibration Monitoring Program (ICMP) method developed by Electric Power Research Institute and other empirical based inferential modeling techniques have been developed but have not become viable options.

Existing solutions to the redundant sensor validation problem have several major flaws that restrict their applications. Parity space method, such as ICMP, are not robust for low redundancy conditions and their operation becomes invalid when there are only two redundant sensors. Empirical based inferential modeling is only valid when intrinsic correlations between predictor variables and response variables remain static during the model training and testing phase. They also commonly produce high variance results and are not the optimal solution to the problem.

This dissertation develops and implements independent component analysis (ICA) for redundant sensor validation. Performance of the ICA algorithm produces sufficiently low residual variance parameter estimates when compared to simple averaging, ICMP, and principal component regression (PCR) techniques. For stationary signals, it can detect and isolate sensor drifts for as few as two redundant sensors. It is fast and can be embedded into a real-time system. This is demonstrated on a water level control system.

Additionally, ICA has been merged with inferential modeling technique such as PCR to reduce the prediction error and spillover effects from data anomalies. ICA is easy to use with, only the window size needing specification.

The effectiveness and robustness of the ICA technique is shown through the use of actual nuclear power plant data. A bootstrap technique is used to estimate the prediction uncertainties and validate its usefulness. Bootstrap uncertainty estimates incorporate uncertainties from both data and the model. Thus, the uncertainty estimation is robust and varies from data set to data set.

The ICA based system is proven to be accurate and robust; however, classical ICA algorithms commonly fail when distributions are multi-modal. This most likely occurs during highly non-stationary transients. This research also developed a unity check technique which indicates such failures and applies other, more robust techniques during transients. For linear trending signals, a rotation transform is found useful while standard averaging techniques are used during general transients.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Background

Empirical sensor validation techniques utilize the redundant information in the system measurements to judge the sensor status. It differs from sensor calibration in that such empirically based judgments often come from modeling the system versus artificially loading the sensor and calibrating it manually. In complex systems such as nuclear power plants, redundancy is an important strategy for maintaining safe and reliable operations. Redundant sensors are installed to increase the reliability of process measurements to assure that the operators and protection systems can maintain the plant in a safe operating condition. This redundant information makes the application of nuclear power plant on-line sensor calibration validation possible.

The birth of current Nuclear Power Plant advanced monitoring and diagnostic techniques can probably be traced to the Three Mile Island (TMI) unit 2 nuclear power plant accident. The TMI-2 accident began with a valve malfunction, which may have been prevented or detected earlier by using current sensor validation and plant monitoring techniques. The negative impact of the TMI-2 accident is very significant. No new nuclear power plant has been ordered in the US since the TMI-2 accident. The TMI-2 accident triggered the first generation of nuclear power plant sensor validation techniques based on parity space methods [EPRI, 1981]. Here, 25 years later, sensor validation technique research is still important and improved solutions are being developed.

The application of soft computing techniques; particularly neural networks, fuzzy logic, and genetic algorithms, to the surveillance, diagnostics and operation of nuclear power plants and their components is an area that still has great potential for exploitation [Uhrig, 1999]. However, practical application of these techniques is uncommon and not an easy task. For instance, many techniques can be shown to work using simulated data, but may not be suitable for actual application due to various non-Gaussian noise realizations and the difficulty to attain training data for off-normal events. Additionally, some techniques, such as those applying neural networks, may suffer from long training times and/or inconsistent results.

Differing from these heuristic approaches, the theory of statistical learning [Vapnik, 1998] tries to approach the problem systematically to find the structure and capacity of a learning machine. Also, the No Free Lunch (NFL) theorem [Wolpert, 1992, 1994, 1995a, 1995b] is derived to answer what cannot be learned from statistical learning processes. The conclusion states that if the training space doesn't completely cover the prediction space, generalization may not produce suitable results. This reminds us that one should be very clear about the fundamental assumptions of empirical learning processes being applied.

1.2 Problem Statement

The redundant measurement of process variables is widely used in safety critical applications such as nuclear power, chemical processing and aerospace industries. Redundant sensors are used to provide independent measurements to guard against sensor channel failures. Additionally, the redundant information can be utilized to consistently

check measurement integrity. This technique is defined as sensor validation or on-line calibration monitoring for redundant sensors. On-line calibration monitoring performs consistence checks automatically during process operation.

Recently, the Office of Nuclear Reactor Regulation Application issued a safety evaluation report (SER) on topical report #104965: "On-Line Monitoring of Instrument Channel Performance". This report focused on the generic application of on-line monitoring techniques to be used as a tool for assessing instrument performance. It proposed to relax the frequency of instrument calibrations required by the Technical Specifications (TS) from once every fuel cycle to once in a maximum of 8 years based on the on-line monitoring results. Implementation of the technique to relax the TS requires a license amendment. The report claims the following benefits:

- Helps eliminate unnecessary field calibrations.
- Reduces associated labor costs.
- Limits personnel radiation exposure.
- Limits potential for miscalibration.

On line calibration monitoring constructs a parameter estimate that is derived from all the redundant sensor measurements. If all the measurements are good and the noise levels are similar, a simple average of the redundant sensors will provide an estimate with a minimum variance. For N channel measurements, the parameter variance will be reduced by a factor of N. However, the simple averaging algorithm is not efficient or robust in detecting a drifting channel due to what is commonly termed spillover. Spillover occurs when a single drifting or faulty sensor affects the parameter estimate. The averaging algorithm is not able to identify the faulty channel when only two

measurement channels are present because the sensor estimate drifts with half the magnitude of the faulty sensor. There are several commercial nuclear power plant applications when only two sensors are available. These commonly occur when the variable is a controlled variable in which an operator must choose which of the two sensors to use for input to the controller. In this case, the simple averaging method cannot be used to identify a faulty sensor.

The Instrument Calibration and Monitoring Program (ICMP) algorithm [Wooten 1993, Davis 1995] is a weighted averaging algorithm that is more robust than simple averaging. It assigns a consistency value to each channel. If all measurement are determined to be consistent, the measurements will be equally weighted and the algorithm is reduced to simple averaging. If one of the measurements begins to drift and differs from the others by more than a stated tolerance, the weight of that measurement will be reduced due to its inconsistency. Thus, the parameter estimate will be less affected by the drifting sensor due to its reduced weight. This algorithm is also not able to detect which of two redundant sensors is drifting.

Simple averaging and ICMP solve the redundant sensor validation problem directly while empirical inferential sensing techniques solve the problem indirectly. The learning process of an empirical sensor validation technique is based on two fundamental aspects: the physics governing the objects to be learned, and the proper mathematics to actually carry out the learning process. Physical foundations provide the scope and existence of a solution. In a nuclear power plant, mass, momentum and energy conservation ensure the existence of correlations between different measurements especially during quasiequilibrium operations. Thus, with measurement of other process

variables, a virtual sensor can be created that provides a reliable estimate of the process variable of interest. This virtual sensor is commonly termed an inferential sensor. The inferential sensor is immune to sensor drift due to the sensor of interest degrading. This results in the residuals between the inferential sensor and the real sensor containing information that can be used to identify a sensor fault. However, empirical-based inferential sensing is an ill-posed problem in which the solution may be unstable due to the collinearity of inputs and the ill-conditioned nature of the process data. Proper regularization techniques are required for successful implementation [Hines, 1999].

Inferential modeling usually employs what is commonly referred to as supervised learning in which the system model is constructed from plant data. The learning process follows two steps. In the first training step, a model is constructed from the training data forming a mapping between predictor (input) variables and the response (output) variables. An objective function is evaluated to measure and optimize the training process. Mean squared error (MSE) is most commonly used as the objective function. In the second step, termed model testing or validation, the model's performance is evaluated using new data, which was not used during training. One needs to prevent overfitting the data during training process, otherwise the generalization error on testing data will be high. Commonly, regularization techniques are used to obtain reliable, repeatable, low variance results.

This dissertation developed, tested, and optimized a new method for redundant sensor calibration validation using an unsupervised learning method: independent component analysis. Additionally, it compares the newly developed method's performance with past techniques.

1.3 Original Contributions

The dissertation develops methods implementing independent component analysis (ICA) for redundant sensor calibration monitoring. ICA is a statistical technique in which the observed data are expressed as a linear transformation of latent variables ('independent components') that are mutually independent [Hyvarinen, 2001]. The ICA method is able to separate mixtures of independent sources in the dataset in order to predict the process parameter more accurately. ICA prediction is very robust in that the faulty component of a sensor measurement does not adversely affect the parameter estimate, even when there are only two redundant sensors.

Independent component analysis is an unsupervised learning paradigm. With supervised learning there is a clear measure of success (i.e., MSE) that can be used to judge adequacy in particular situations and to compare the effectiveness of different methods over various situations. In the context of unsupervised learning, there is no such direct measure of success. One must judge the results heuristically and more carefully [Hastie, 2001]. For the redundant sensor validation problem, we have *a prior* information to use to ease the judgment. This leads to the use of a reliability module in the ICA method for redundant sensor validation.

Moreover, inferential modeling can detect common mode failures while independent component analysis provides channel estimate with less noise which can detect the drift faster. This research will investigate the merging of inferential modeling with independent component analysis for validating redundant sensors.

The original contributions of the dissertation are as follows:

- The use of independent component analysis for redundant sensor validation that can monitor systems with as few as two redundant sensors.
- The development of a reliability monitoring module using unity check for robust parameter estimation.
- The demonstration of a real time controller utilizing independent component analysis.
- The use of a rotation transform for linear nonstationary signals.
- The development of a system that merges inferential modeling with independent component analysis to enhance the robustness and reduce uncertainties for redundant sensor validation.
- The use of bootstrap for constructing prediction intervals for hybrid inferential modeling and the ICA method.

1.4 Organization of the Dissertation

The dissertation is organized as follows. Chapter 2 is a literature survey of current methods for redundant sensor validation such as parity space methods, the Instrumentation and Calibration Monitoring Program (ICMP) and empirical based inferential modeling. Chapter 3 provides the theoretical foundations of independent component analysis. A solution for scaling and permutation problem is also provided and a unity check for ICA robust monitoring is derived. Chapter 4 contains the results. First, a toy problem is solved with independent component analysis. Second, ICA is applied to

real nuclear power plant data for sensor monitoring and its results are compared with ICMP results. Robustness is tested by bootstrapping and noise reduction characteristics are presented. A linear relationship between the ICA prediction and measurement variance termed the selection rule is also developed. Next, a water tank level experiment demonstrates a real time application of ICA. It also shows the limitation of classic ICA on nonstationary signals. The reliability monitoring module guards the ICA solution from going out of the solution space. If the unity check does not pass, the ICA prediction automatically degrades to a reliable simple average. For a linear trending nonstationary signal, a rotation transform can be applied. The results of the ICA controller are also discussed. A hybrid method merging inferential modeling and ICA is applied to both drifting data and data containing nuclear power plant abnormalities. Chapter 5 develops an uncertainty analysis utilizing the bootstrap method. First, the bootstrap method for prediction interval estimation is reviewed. Next, a prediction interval for a least squares model is calculated. Last, the bootstrap prediction intervals are calculated for the ICA and PCR hybrid model. Chapter 6 concludes this dissertation and provides recommendations for future work. A summary of published articles based on this dissertation and Matlab code used for calculation is included in the appendix.

CHAPTER 2
LITERATURE SURVEY

A review of the literature on redundant instrument channel monitoring indicated that there are three main approaches that have been developed:

1. Parity Space methods
2. Instrumentation and Calibration Monitoring Program (ICMP) method
3. Empirical based methods

Much of the early work in sensor validation monitoring has been centered on Fault Detection and Isolation (FDI) techniques.

2.1 Parity Space Approach

This section will review the parity space approach and present its theoretical derivation.

2.1.1 Review

Parity space approaches are rooted in the FDI schemes. FDI techniques [Isermann 1997] for redundant instrument channels are often model-based methods [Frank 1990, Hall 1983, Jin 1997, Willsky 1976], or knowledge-based methods [Benkhedda 1996, Frank 1990]. Neural Networks have also been employed [Chowdhury 1996, Chan 1998], as well as a recently proposed Constrained Kohonen Network approach [Chan 2001].

Parity space method requires three or more redundancies in the sensor measurements. When there are only two redundant sensors, direct comparison is not

possible. One can use analytic redundancy [Ray 1983a, Simani, 2000, Desai 1979, Frank 1990, Venkateswaran 2002, Patton 1997] to serve as an additional sensor.

The parity space approach is a very popular fault detection and isolation technique for use with redundant sensors. The parity-space technique models the redundant measurements as the true value combined with a measurements error term. Measurement inconsistencies are obtained by eliminating the underlying measured quantity from the data by creating linearly independent relationships between the measurements known as the parity equations. The magnitude of the parity vector represents the inconsistency between the redundant measurements, and the direction of the parity vector indicates the faulty signal. Patton and Chen reviewed redundancy management in fault detection with emphasis on the parity space approach [1991]. The parity space method is a two-stage process: (1) residual generation and (2) decision making. Residual generation has been studied by many researchers [Frank 1997, Basseville 1997, Frisk 2001, Krishnaswami 1995, Nyberg 1997, Gertler 1995, Gertler 1997], and was included under redundant channel monitoring work as reported by Ray, et. al. [1983a, 1983b, 1986, 2000]. The decision making stage has been approached in a number of ways including the sequential probability ratio test [Ray 1989, 1991], and an approach via multiple hypothesis testing [Ray 2002]. Without a third signal from analytical redundancy, parity space approaches cannot detect faults in the two-channel condition.

2.1.2 Parity Space Approach Redundancy Management Procedure

Theory of the redundancy management procedure is modeled by (2.1) [Ray 1986].

$$z(t) = [H(t) + \Delta H(t)]x(t) + \beta(t) + e(t) \qquad (2.1)$$

where: z is an $(l \times 1)$ vector of known measurements

H is an $(l \times n)$ a priori known measurement matrix of rank n $(l > n)$

ΔH is an $(l \times n)$ matrix representing unknown scale factor errors

x is the $(n \times 1)$ unknown vector variable that is to be estimated

β is an $(l \times 1)$ unknown vector of bias errors

e is an $(l \times 1)$ vector of measurement noises with $E(e) = 0$

The effects of the scale factor errors and the bias errors can be combined [Ray 1984]:

$$c(t) = \Delta H(t)x(t) + \beta(t) \tag{2.2}$$

Calibrated measurements (unfaulted situation) are defined as:

$$m(t) = z(t) - \hat{c}(t) = H(t)x(t) + e(t) \tag{2.3}$$

where: $\hat{c}(t)$ is the estimate of $c(t)$

$\varepsilon(t) = (c(t) - \hat{c}(t)) + e(t))$ is the additive noise and remaining error associated with the calibrated measurements

Rewriting the equation for the calibrated measurements, dropping the time dependency:

$$m = Hx + \varepsilon \tag{2.4}$$

where: m is an $l \times 1$ vector of process measurements

H is the measurement matrix

x is the true value of the process variable

ε is the measurement noise, such that for normal functioning of each measurement, $| \varepsilon_i | \leq b_i$. Where b_i is the error bound of the i^{th} sensor, $i = 1,2,...,l$.

For scalar sensors, the measurement matrix can be chosen as $H = [1 \quad 1 \quad \cdot \quad \cdot \quad \cdot \quad 1]^T$.

Any two measurements are declared consistent at sampling instant k if:

$$| m_i(k) - m_j(k) | \leq b_i(k) + b_j(k) \qquad i = 1,2,...,l \text{ , and } j = 1,2,...,l \quad (2.5)$$

Consistency can be determined instantaneously, or via sequential testing to incorporate prior observations. Sequential testing should limit the occasional inconsistency under normal operating conditions, and thus reduce the probability of false alarms.

The consistencies among the measurements should be independent of x. The variations in the underlying variable x can be eliminated by projecting the vector of process measurements, m, onto the left null space of the measurement matrix, leaving only the effects of the noise vector ε. The projection of m on to the parity space of dimension $(l-1)$, known as the parity vector, is given by:

$$p = Vm = V\varepsilon \qquad (2.6)$$

V is chosen so that its rows form an orthonormal basis for the parity space:

$$VH = 0 \qquad VV^T = I_{l-1} \qquad V^TV = I_l - H(H^TH)^{-1}H^T \qquad (2.7)$$

Under normal conditions when all sensors are operating properly, the parity vector p is small. If a failure occurs, the parity vector grows in magnitude in the direction associated with the failed measurements. The increased magnitude of the parity vector indicates a failure, and the direction of the magnitude increase can be used to identify the failed measurement.

This redundancy management procedure has been further enhanced in a recent publication [Ray 2000]. The enhancements have led to a calibration and estimation filter for redundancy management of sensor data that has the following features:

- All signals are simultaneously calibrated on-line to compensate for their relative errors.
- The weights of individual signals for computation of a least-square estimate of the measured variable are adaptively updated as functions of the respective a posteriori probabilities of failure.

The algorithm carries out FDI whereby for an abrupt change in a redundant signal in excess of its allowable bound, the respective signal is isolated and only the remaining signals are calibrated to provide an unbiased estimate of the measured variable. For a gradual degradation (drift), the influence of the faulty signal is diminished as a function of its deviation from the remaining signals. This is achieved by decreasing the relative weight of the drifting signal as its deviation from the estimate increases. Additional enhancements have been added for multi-level hypothesis testing [Ray 2002]. Multi-level hypothesis testing provides a more precise characterization of potential faults than the bi-level fail/no-fail hypothesis testing, and is often essential for early warning of drifting instrument channels.

2.2 Instrument Calibration and Monitoring Program (ICMP)

The Instrument Calibration and Monitoring Program (ICMP) algorithm is a simplified version of the parity space approach. The basis for the ICMP algorithm has been rigorously developed and its implementation has been tested [Wooten 1993, Davis 1995]. The ICMP algorithm produces a parameter estimate using a weighted average. Weighting is performed based on the consistency values assigned to each redundant channel. These consistency values are updated upon each new observation. At the

limits, if all measurements are declared consistent, the parameter estimate is the simple average. If all measurements are declared inconsistent, no parameter estimate is produced. After obtaining a parameter estimate, individual deviations between each channel measurement and the estimate are computed and compared against specified acceptance criteria. Deviations exceeding the acceptance criteria indicate a faulty sensor. ICMP cannot identify faults in two-channel case.

The ICMP algorithm [EPRI, 2000] calculates a parameter estimate based on a weighted average of a set of redundant sensors. Weighting of individual sensors is based on their consistency with the other sensors in the group. Consistency is evaluated as the absolute difference between a given sensor, and the other sensors in the group. The consistency value ranges from 0 to $n-1$, where n is the number of sensors in the group. Each sensor is assigned a consistency for each data sample evaluated. A sensor's consistency is calculated as follows:

$C_i = 0$

If $\left|m_i - m_j\right| \le \delta_i + \delta_j$, then $C_i = C_i + 1$ (2.8)

where: $C_i =$ the consistency value of the ith signal

$m_i =$ the output for signal i

$m_j =$ the output for signal j

$\delta_i =$ the consistency check allowance for instrument i

$\delta_j =$ the consistency check allowance for instrument j

The values for the consistency check allowances are dependent on the uncertainty present in the signals such as.

$$\left|m_i - m_j\right| \leq 2\delta \tag{2.9}$$

After the consistency values are calculated, the ICMP parameter estimate can be calculated as:

$$\hat{x} = \frac{\sum_{i=1}^{n} w_i C_i m_i}{\sum_{i=1}^{n} w_i C_i} \tag{2.10}$$

where: $\hat{x}$ = the ICMP parameter estimate for the given data sample

w_i = the weight associated with the i^{th} signal

The weight values are included to allow the user to apply a greater weighting to more accurate or reliable sensors within a redundant group. If there is no preference within the group, all weight values can be set to 1, reducing the equation to:

$$\hat{x} = \frac{\sum_{i=1}^{n} C_i m_i}{\sum_{i=1}^{n} C_i} \tag{2.11}$$

The consistency check factor controls the influence of an individual signal on the ICMP parameter estimate. If all sensors are considered equally consistent, then the ICMP estimate is just the simple average of the redundant sensors. If a sensor's consistency value is zero, then it will not influence the parameter estimate.

Once the parameter estimate is calculated, the ICMP algorithm evaluates the performance of each individual sensor relative to the parameter estimate. This is done through the use of an acceptance criterion.

If $|\hat{x} - m_i| \geq \alpha_i$, then m_i has potentially drifted beyond desired limits.

where: α_i = the acceptance criterion for the i^{th} signal

When the deviation between a sensor's measruement and the current parameter estimate exceeds the acceptance criterion, that sensor is considered to have drifted out of calibration. At this point the sensor is assumed to have failed. Note that failing the acceptance criterion does not necessarily disallow the failed sensors value to influence the ICMP estimate. The consistency check factor must also be exceeded, and it is not indirectly related to the acceptance criterion.

2.3 Empirical Based Methods

Researchers at University of Tennessee have been pioneers in the inferential modeling approach for on-line sensor calibration verification systems. Dr. Upadhyaya was one of the original investigators, in the 1980's [Upadhyaya, 1985], to investigate the application of artificial intelligence techniques to nuclear power plants. In the early 1990's, Dr. Uhrig continued this research using neural network techniques [Ikonomopoulos, Uhrig 1992]. Major inferential modeling techniques used by researchers including autoassociative neural networks [Upadhyaya 1992, Hines 1998, Fantoni 1998], principal component analysis [Qin 1999], non-linear partial least squares [Qin 1992, Rasmussen 2000], and kernel based techniques such as Multivariate State Estimation System (MSET) [Gross 1997]. A comparison of the three major techniques: Autoassociative Neural Networks,

Multivariable State Estimation Technique, and Non-Linear Partial Least Squares is given by Hines [2001].

In order to get consistent, repeatable, and quantifiable results, regularization techniques [Hines 1999, 2000, Gribok, 2000, 2001] should be applied.

2.3.1 Ordinary Least Squares Regression

Ordinary least squares regression is a basic algorithm for illustrating inferential techniques. It is also useful for performance bench marketing. Its derivation can be found in [Hastie, 2001] and it is restated here.

Given a linear model

$$y = X\beta + \varepsilon \tag{2.12}$$

where: y is a measurement with N observations

X is p vector of inputs, each vector has N observations

$\varepsilon \sim N(0, \sigma^2)$ is the noise

Estimate the coefficients $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_p)^T$ to minimize the residual sum of squares

$$RSS(\hat{\beta}) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{N} (y_i - \hat{\beta}_0 - \sum_{j=1}^{p} x_{ij}\hat{\beta}_j)^2 \tag{2.13}$$

In matrix form, denote by X the N x (p+1) matrix with each row an input vector (with a 1 in the first position), and similarly let y be the N-vector of outputs

$$RSS(\hat{\beta}) = (y - X\hat{\beta})^T (y - X\hat{\beta}) \tag{2.14}$$

The solution is derived by setting derivative of the cost function to zero.

$$\frac{\partial RSS}{\partial \hat{\beta}} = -2X^T (y - X\hat{\beta}) \tag{2.15}$$

$$\frac{\partial^2 RSS}{\partial\hat{\beta}\partial\hat{\beta}^T} = -2X^T X \tag{2.16}$$

Assuming that X is nonsingular and hence X^TX is positive definite, set the first derivative to zero to get minimum value.

$$X^T(y - X\hat{\beta}) = 0 \tag{2.17}$$

to obtain the unique solution

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{2.18}$$

The predicted values at the training inputs are

$$\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y \tag{2.19}$$

The matrix $H = X(X^T X)^{-1} X^T$ is called "hat" matrix because it puts the hat on y. Parameter uncertainties can be estimated using an estimate of the noise ε.

$$Var(\hat{\beta}) = (X^T X)^{-1}\sigma^2 \tag{2.20}$$

$$\hat{\sigma}^2 = \frac{1}{N - p - 1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.21}$$

where: N are the number of observations.

p are the number of inputs.

Assume the linear model is correct and the deviations of Y around its expectation are additive and Gaussian.

$$Y = \beta_0 + \sum_{j=1}^{p} X_j\beta_j + \varepsilon \tag{2.22}$$

$$\varepsilon \sim N(0,\sigma^2) \tag{2.23}$$

Then,

$$\hat{\beta} \sim N(\beta,(X^T X)^{-1}\sigma^2) \tag{2.24}$$

$$(N-p-1)\hat{\sigma}^2 \sim \sigma^2 \chi^2{}_{N-p-1}$$

$\hat{\beta}$ and $\hat{\sigma}$ are statistically independent. We use these distributional properties to form tests of hypothesis and confidence intervals for the parameters.

Z-score is

$$z_j = \frac{\hat{\beta}_j}{\hat{\sigma}\sqrt{v_j}}$$ where v_j is the jth diagonal element of $(X^T X)^{-1}$. (2.25)

1-2α confidence interval for β_j is

$$(\hat{\beta}_j - z^{(1-\alpha)} v_j \hat{\sigma}, \hat{\beta}_j + z^{(1-\alpha)} v_j \hat{\sigma}) \tag{2.26}$$

Here $z^{(1-\alpha)}$ is the 1-α percentile of the normal distribution.

For example, for α=0.025, $z^{(1-\alpha)}$ =1.96.

The confidence set for the entire parameter vector β is

$$C_\beta = \{\beta \mid (\hat{\beta}-\beta)^T X^T X(\hat{\beta}-\beta) \le \hat{\sigma}^2 \chi^2_{p+1}{}^{(1-\alpha)}\} \tag{2.27}$$

The mean squared error of an estimator $\hat{y}$ in estimating y is

$$MSE(\hat{y}) = E(\hat{y}-y)^2 = Var(\hat{y}) + [E(\hat{y})-y]^2 \tag{2.28}$$

Least squares estimates have the smallest variance among all linear unbiased estimates. However, there may well exist a biased estimator with smaller mean squared error. Such

an estimator would trade a little bias for a larger reduction in variance. Biased estimators, such as ridge regression, are commonly used. Any method that shrinks or sets to zero some of the least squares coefficients may result in a biased estimate.

2.3.2 Principal Component Analysis

Principal Component Analysis (PCA) has been frequently reported in the literature for redundant channel monitoring applications. An excellent introduction to PCA, and its applications, is provided by I.T. Joliffe [1986] in his book *Principal Component Analysis*.

There are a variety of ways the principal components can be exploited to monitor changes in an on-line monitoring approach. The statistics of the principal components themselves, or the relationships between principal components, can be monitored for changes [Amand 2001, Doymaz 2001a, Kano 2000, 2001, Seiter 2001]. Many modifications to *traditional* PCA have also been reported [Doymaz 2001b, Vigneau 2002]. Other approaches have used principal components for residual generation within the framework of the traditional parity space approach [Dunia 1998a].

2.3.3 PCA for Redundant Sensor Monitoring

The basic PCA approach linearly transforms a data matrix, $\mathbf{X}$ of p columns (sensors) and n rows (observations), to an orthogonal principal components space of equivalent dimensions [Rasmussen, 2002; Ding, 2004] . The transformation occurs such that the direction of the first principal component is determined to capture the maximum variation of the original data set. The variance of subsequent principal components is the

maximum available in an orthogonal direction to all previously determined principal components. The full set of principal components is an exact copy of the original data set, though the axes have been rotated. Selecting a reduced subset of components results in a reduced dimension structure with the majority of the information available, in which information is assumed to be equivalent to variance. Usually, small variance components that are not retained are assumed to contain unrelated information such as process or measurement noise. In the context of redundant instrument channel monitoring, it is assumed that the first principal component can provide an estimate of the true measured parameter, and the smaller variance components are assumed to be negligible with respect to the estimation.

To avoid ambiguities in the terminology, the set of orthogonal components derived from PCA, which are sometimes referred to as principal component scores, will be referred to herein as principal components and denoted in matrix form as $\mathbf{Z}$. The transformation of the data matrix to the principal component matrix is given by:

$$\mathbf{Z} = \mathbf{XA} \tag{2.29}$$

where: $\mathbf{Z}$ is the $(n \times p)$ principal component matrix

$\mathbf{X}$ is the $(n \times p)$ matrix of *mean-centered* measurement data with *n* observations and *p* redundant sensors

$\mathbf{A}$ is the $(p \times p)$ matrix of eigenvectors of $\mathbf{X}^T\mathbf{X}$

The following notation is also defined:

$\mathbf{a}_j$ is the j^{th} eigenvector of $\mathbf{X}^T\mathbf{X}$, where $j = 1,..., p$, of dimensions $(p \times 1)$

$\mathbf{z}_j$ is the j^{th} principal component, where $j = 1,..., p$, of dimensions $(n \times 1)$

z_{ij} is the scalar value of the j^{th} principal component corresponding to the i^{th} $(i = 1,...,n)$ redundant measurement vector $\mathbf{x}_{[i,:]} = [x_{i1} \quad x_{i2} \quad \cdot \quad \cdot \quad \cdot \quad x_{ip}]$

Due to the reasonable assumption that a set of redundant instrument channel measurements can be approximated by the first, or primary, principal component; an estimate of the *mean-centered* true value of the process at the i^{th} sample is given by:

$$p_i = z_{ij} = \mathbf{x}_{[i,:]}\mathbf{a}_1 \tag{2.30}$$

For the full data matrix $\mathbf{X}$, we can write:

$$\mathbf{p} = \mathbf{z}_1 = \mathbf{X}\mathbf{a}_1 \tag{2.31}$$

where: p_i is a scalar *mean-centered* estimate based on $\mathbf{x}_{[i,:]}$

$\mathbf{p}$ is a vector *mean-centered* estimate based on $\mathbf{X}$, of dimensions $(n \times 1)$

Finally, we define the parameter estimate after mean-centered scaling, in scalar and vector form, as:

$$p_i^f = p_i + m_{\mathbf{X}}^w \qquad \mathbf{p}^f = \mathbf{p} + \begin{bmatrix} 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1_n \end{bmatrix} \cdot m_{\mathbf{X}}^w \tag{2.32}$$

where: $m_{\mathbf{X}}^w = \left[\mathbf{a}_1^T \cdot \begin{bmatrix} \overline{\mathbf{x}}_{[:,1]} \\ \overline{\mathbf{x}}_{[:,2]} \\ \cdot \\ \cdot \\ \cdot \\ \overline{\mathbf{x}}_{[:,p]} \end{bmatrix} \right] \cdot \dfrac{1}{\sum_{j=1}^{p} a_{j1}}$ **(**the eigenvector weighted mean value**)**

$$\bar{\mathbf{x}}_{[:,p]} = n^{-1}\sum_{i=1}^{n} x_{[i,p]} \qquad \text{(the mean value of the } p^{\text{th}} \text{ channel)}$$

An obvious result is that if the channels are identical, the first principal component will be an equal combination of each channel, i.e. if:

$$\mathbf{x}_{[:,1]} = \mathbf{x}_{[:,2]} = \cdots = \mathbf{x}_{[:,p]}, \text{ then } a_{11} = a_{21} = \cdots a_{p1} = \sqrt{\frac{1}{p}}. \qquad (2.33)$$

In general terms, if an input channel's variance is higher than the remaining channels, its corresponding element in the eigenvector of the first principal component will also be higher. Thus, the influence of a particular channel's measurements on the final parameter estimate can be related to that channel's sample variance.

In an on-line monitoring approach, a set of measurement data would be used to calculate the eigenvector, $\mathbf{a}_1$, the mean values of the redundant channels, and the weighted mean value, $m_{\mathbf{X}}^{w}$. Subsequent estimations could then be produced based on future observations via:

$$p_k^f = \mathbf{x}_{[k,:]}\mathbf{a}_1 + m_{\mathbf{X}}^{w} \qquad (2.34)$$

where: $\mathbf{x}_{[k,:]}$ is a future observation vector

p_k^f is the corresponding estimate

In applying the PCA method for redundant instrument channel monitoring, it is important to keep in the mind the assumptions being made. It is assumed that a sufficient parameter estimate is available in the first principal component, and that the variance of the redundant channel measurement vectors is related to the overall process in such a way

that higher variant measurements should be allowed a greater influence on the parameter estimate. Due to the underlying importance of the sample variance calculations for this method, it should be verified that the data being used are free of outliers.

The residuals between the resultant parameter estimate, shown in equation 2.34, and each redundant channel measurement are then evaluated for each observation in an on-line mode, or alternatively in a batch mode. Sensor drifts are suspected when a channel's residual deviates from some nominal value determined during the calculation of the model parameters using a representative data set.

The principal components of correlated sensors can be used to produce estimates of the sensors via Principal Component Regression (PCR), [Marbach 1990, Dunia 1998b, Martens 1989]. This is termed inferential modeling and is discussed in Chapter 3.

2.4 Independent Component Analysis

Inferential modeling belongs to supervised learning paradigm. Learning is based on a training dataset. Independent component analysis (ICA) resolves a data structure directly based on the "independence" of different sources. ICA is used as a redundant sensor validation tool in this dissertation.

2.4.1 Background on ICA

Independent Component Analysis (ICA) was introduced in the early 1980s and attained wide attention and growing interest in the mid 1990s. The technique attempts to identify original signals from a mixture of observed signals, which are a linear combination of sources, without knowing any information about the mixing matrix, or

having any prior information about the sources except that they are assumed to be independent and have non-Gaussian distributions. Since the sources are assumed to be independent they are termed independent components (ICs). One requirement for isolation of the ICs is that only one component can have a Gaussian distribution. A detailed survey of ICA can be found in *Neural Computing Surveys* [Hyvarinen, 1999b]. Hyvarinen, et al, [2001] also wrote a good introductory book on ICA.

ICA is rooted from the need to find a suitable linear representation of a random variable. The classic method to solve this problem is to use second order information in the covariance matrix such as principal component analysis and factor analysis [Harman, 1967; Jolliffe, 1986; Kendall, 1975]. Rather than applying independence as a guiding principle, PCA attempts to linearly transform a data set resulting in uncorrelated variables with minimal loss of information [Hyvarinen 2001]. For Gaussian distributed variables, uncorrelatedness is identical to independence. For non-Gaussian distributed variables, independence is a much stronger requirement than uncorrelatedness. For non-Gaussian data, higher order statistics are needed to obtain a meaningful representation. Projection pursuit is a technique for finding interesting features of data such as clusters using higher order statistics [Friedman, 1974, 1987; Huber, 1985; Jones, 1987; Sun, 1993; Cook, 1993]. Projection pursuit uses a cost function such as differential entropy [Huber, 1985; Jones, 1987] rather than the mean-squared error used in the PCA transformation. Projection pursuit is interested in the non-Gaussianity of data. There are connections between ICA and these techniques. In the noise free case, ICA is a special case of projection pursuit. ICA also can be viewed as a non-Gaussian factor analysis. ICA must use higher order statistics while PCA only uses second order statistics.

The properties of the ICA method [Hyvarinen, 1999b] is restated here:

- " The statistical properties (e.g. consistency, asymptotic variance, robustness) of the ICA method depend on the choice of the objective function."
- " The algorithmic properties (e.g., convergence speed, memory requirements, numerical stability) depend on the optimization algorithm."

The ICA algorithm is described in detail in chapter 3. As a research application, the first property is investigated intensively in this work.

2.4.2 ICA Applications

ICA has found many successful applications in blind source separation [Jutten, C., 1991], telecommunication, imaging processing (feature extraction), and brain imaging applications (EEG and MEG) [Lee 1998, Roberts 2001]. More applications are seen in:

- Stock portfolio selection [Back, A.D., 1997].
- Audio separation [Torkkola, 1999].
- Text document analysis [Isbell, 1999].
- Rotating machine monitoring [Ypma, 1999].
- Seismic monitoring [Ham, 1999].
- Reflection canceling [Farid, 1999].
- Nuclear magnetic resonance spectroscopy [Nuzillard, 1998].

ICA for sensor validation was first developed by the author [Ding 2003a, 2003b, 2004] and his colleagues at the University of Tennessee.

However, ICA is still not a mature technique. Many practical problems beyond the basic ICA model exist. Some of these include the nonlinear mixing problem [Burel

1992, Lee 1997, Taleb and Jutten 1997, Yang 1997], underdetermined ICA model, i.e, more sources than sensors [Lewicki and Sejnowski, 2000], noisy ICA [Nadal, 1994; Attias, 1999] and non-stationary problem [Matsuoka, 1995; Murata, 1997; Perra, 2000]. For a nonstationary time structure, alternative assumptions other than non-Gaussianity of ICs are needed. One can use different autocovariance functions [Molgedey, 1994] or nonstationary variances [Matsuoka, 1995] for a successful separation.

The user should be very cautious at the application of statistical learning methods by fully understanding the basic underlying assumptions for any particular applications. This is also true for sensor validation using ICA. A major work of this dissertation focuses on the validation of these assumptions for the redundant sensor monitoring task.

In the next chapter, the method of classic ICA is discussed. For the task of redundant sensor validation, *a prior* information is used to identify a proper independent component. Moreover, a unity check is derived for ICA reliability monitoring.

CHAPTER 3
METHODOLOGY

3.1 Information Theory Preliminaries

The ICA model can be understood more easily from the viewpoint of information theory [Deco, 1996]. The concept of entropy is the basis of the information theory. For a given random variable X, with probability density p(x), the entropy H(X), is defined by Shannon [1948] :

$$H(X) = -\sum (p(x) \cdot \log(p(x))) = E(\log(\frac{1}{p(x)})) \qquad (3.1)$$

where E(.) denotes the expectation operator. Entropy gives a measure of randomness for the given distribution. The measure of entropy for a rare event will be large and so a large amount of information is due to this event. For a continuous variable, the differential entropy is introduced by changing the summation to an integral [Girolami, 1999]. Consider a discrete histogram:

$$H = -\lim_{N_B \to \infty} \sum_{i=1}^{N_B} p(x_i)\Delta \log(p(x_i)\Delta) \qquad (3.2)$$

for a continuous pdf that $\int p(x)dx = 1$ then the entropy is

$$H = -\int p(x)\log(p(x))dx + \lim_{N_B s \to \infty} \log(\Delta) \qquad (3.3)$$

Omit the constant, differential entropy of a continuous random variable is

$$H(x) = -\int p(x)\log(p(x))dx \qquad (3.4)$$

For two distribution p(x) and q(x), the Kullback-Leibler entropy [Deco, 1996] is a measure of the difference between the two distributions and defined as:

$$K(p,q) = \sum p(x)\log(\frac{p(x)}{q(x)}) \qquad (3.5)$$

Lemma 3.1.1: Entropy of uniform distribution [Deco, 1996]

If a continuous random variable is uniformly distributed between 0 and a, its entropy is given by:

$$H(x) = \log(a) \qquad (3.6)$$

Lemma 3.1.2: Entropy of normal distribution [Deco, 1996]

For a normal distribution defined as:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-x^2}{2\sigma^2}},$$

the corresponding entropy is:

$$H(x) = \frac{1}{2}\ln(2\pi e\sigma^2) \qquad (3.7)$$

Theorem 3.1.1: Gibbs second theorem [Deco, 1996]

Let $X_1, X_2, \ldots X_k$ be a vector of random variables distributed according to the density $p(x_1,x_2,\ldots x_k)$ with zero mean and covariance matrix C, then

$$H(X_1,\ldots X_k) \le \frac{1}{2}\log\left\{(2\pi e)^k \|C\|\right\} \qquad (3.8)$$

This theorem shows that the normal distribution maximizes the entropy over all distributions with the same covariance matrix.

3.2 ICA Model and Algorithm

Independent component analysis is a statistical model in which the observed data (X) is expressed as a linear transformation of latent variables ('independent components', S) that are non-Gaussian and mutually independent. We may express the model as

$$X = \mathbf{A}\, S \qquad (3.9)$$

where: X is an (n x p) data matrix of n observations from p sensors

S is an (p x n) matrix of p independent components

A is an (n x p) matrix of unknown constants, called the mixing matrix

The problem is to determine a constant (weight) matrix, **W,** so that the linear transformation of the observed variables

$$Y = \mathbf{W}\, X \qquad (3.10)$$

has some suitable properties. In the ICA method, the basic goal in determining the transformation is to find a representation in which the transformed components, y_i are as statistically independent from each other as possible. When random variables with

specific non-Gaussian distributions are combined, the central limit theorem shows that the sum is more Gaussian than the original variables. Therefore, to separate the original variables (S) from a sum (X), we want to choose a transformation (W) that makes them as non-Gaussian as possible. We then assume that the maximally non-Gaussian signals Y are estimates of the original independent components, one of which is the parameter value, and thus the parameter estimate.

Hyvarinen [1999], developed an ICA algorithm called FastICA as described below. It uses negentropy J(y) as the measurement of the non-Gaussianity of the components.

$$J(y) = H(y_{gauss}) - H(y) \qquad (3.11)$$

H(y) is the differential entropy of a random vector *y*.

$$H(y) = -\int f(y) \log f(y) dy \qquad (3.12)$$

where: *f(y)* is the density of the random vector *y*.

Based on the maximum entropy principle, negentropy J(y) can be estimated:

$$J(y_i) \approx c[E\{G(y_i)\} - E\{G(\nu)\}]^2 \qquad (3.13)$$

where: G is any nonquadratic function

c is an irrelevant constant

ν is a Gaussian variable of zero mean and unit variance

E{} is the expectation

One attempts to maximize negentropy so that a non-linear transformation of y is as far as possible from a nonlinear transformation of a Gaussian variable (v). This nonlinear transformation (G) is also called a contrast function. The following is a commonly used contrast function G and its derivative g:

$$G(u) = \frac{1}{a_1} \cdot \log\cosh(a_1 u)$$
$$g(u) = \tanh(a_1 u) \qquad (3.14)$$

The FastICA algorithm for estimating several independent components is described below:

1. Center the data to make its mean zero.
2. Whiten the data to give **z** with unit variance.
3. Choose *m*, the number of independent components to estimate.
4. Choose initial values for the $\mathbf{w}_i$, i=1,…,m, each of unit norm. Orthogonalize the matrix **W** as in step 6 below.
5. For every i=1,…,m, let $\mathbf{w}_i \leftarrow E\{\mathbf{z}G(\mathbf{w}^T_i\mathbf{z})\}-E\{g(\mathbf{w}^T_i\mathbf{z})\}\mathbf{w}$, where G and g are defined in (2.14)
6. Perform a symmetric orthogonalization of the matrix $\mathbf{W} = (\mathbf{w_1}, \ldots \mathbf{w_m})^T$ by $\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^T)^{-1/2}\mathbf{W}$.
7. If matrix has not converged, go to step 5.

A concern with using ICA is that it has two ambiguities [Hyvarinen 2001]. One is that the variances (energies) of the independent components cannot be determined. The other is that the order of the independent components cannot be determined. These ambiguities are of concern when performing on-line instrument channel monitoring for two reasons: 1). the components must be scaled back to their original units and 2). the component containing the parameter estimate needs to be selected.

In order to scale the components back to their original units, we need to calculate the correct scale factor α by selecting the component corresponding to the parameter of

interest. To do this, the mean of the measured parameter is estimated by taking the mean of the medians for each (*i*) of n channels (see equation 3.12). Next, compute n scale factors by dividing the mean of the parameter by the mean of each component and use those scale factors to give the components the same mean as the measured parameter. The scaled component with the highest correlation coefficient to the raw signals (X) is the component of interest and is the parameter estimate.

$$\alpha_i = \frac{mean(median(X))}{median(IC_i)} \qquad i=1\ldots n \qquad (3.15)$$

where: X is the matrix of n mixed signals

IC_i is the i^{th} independent component

mean and median are MATLAB functions

To calculate the correct transformation matrix, rescale the transformation matrix **W** to $\mathbf{W_c}$:

$$W_c = \text{sign}(\alpha) * \alpha * W \qquad (3.16)$$

where: α is the α_i that maximizes the correlation between the scaled component and the parameter value

The parameter estimate is now calculated with:

$$Y = \mathbf{W_c}\, X \qquad (3.17)$$

The residuals between this parameter estimate and the channel measurements are evaluated to assess the calibration status of the redundant instrument channel sensors. Sensor drifts are suspected when a channel's residual deviates from some nominal value determined using a representative data set.

As stated by Perra [2000], and validated by our own research, many ICA methods are shown to work satisfactory in computer simulations but perform poorly in real environments. One reason is the actual signals found in practice are commonly nonstationary. For these cases, other separation principles should be used. Since U.S. nuclear power plant signals are usually stationary, the ICA algorithm is used in conjunction with a simple method to detect any non-stationary conditions called a unity check.

For the model,

$$y = \sum_{i=1}^{n} w_i x_i \tag{3.18}$$

Taking the expectation of both sides of equation 3.18 results in:

$$E(\hat{y}) = \sum_{i=1}^{n} w_i \cdot E(x_i)\,. \tag{3.19}$$

Since x_i are redundant sensor measurements and $\hat{y}$ is the parameter estimation, the expectations would give the same value resulting in:

$$\sum_{i=1}^{n} w_i = 1\,. \tag{3.20}$$

Since there are uncertainties in the estimation, a tolerance check is used:

$$\left| \sum_{i=1}^{n} w_i \right| - 1 < \varepsilon \tag{3.21}$$

where ε is a measure of uncertainty. For ε = 0.05, there is 10% uncertainty in the estimation.

The unity check uses *a prior* information. It is used to determine whether the ICA algorithm is operating correctly and indicates when it fails: such as when the signals are non-stationary. If ICA fails to pass the unity check, direct averaging is used for parameter estimation. This results in the ICA algorithm being used when the signal is stationary and direct averaging being used during transients.

3.3 Model Justification and Drift Detection

The measurements from each channel contain the process parameter, a common noise source and independent channel noise sources. These three components are most likely independent from each other. Except for the channel noise, the other two components are seldom a Gaussian distribution. Another assumption is that the transform matrix **A** is linear and time invariant. These assumptions are valid at most conditions, but are especially valid for fairly steady state measurements. Since U.S. nuclear power plants operate most of the time at nearly 100% and steady state, this method is valid. Even when the plant is not operating at steady state, more than one source is probably not Gaussian, and if they are, the method degrades towards simple averaging. Recall that noise sources are commonly assumed to be Gaussian because when noise sources are added they tend towards Gaussian; this does not imply the original sources are Gaussian.

Moreover, during faulty conditions, the fault component is introduced into one or more redundant channels and is absolutely independent from the process parameter. Therefore, we can create the model not only from fault free data (regression), but we can also build the model using data when the drift is present because of the model's ability to separate the independent components.

A given channel's residuals are defined as the differences between the parameter estimate and the channel measurements. Each channel will have a unique residual with respect to the parameter estimate, which is an estimate of the independent channel noise source: the channel drift. The mean values and standard deviations of the residuals will be used to identify out-of-calibration channels via the following rule:

If $\bar{\mathbf{r}}_j - 2\sigma_j \le p_k^f - x_{kj} \le \bar{\mathbf{r}}_j + 2\sigma_j$, then the j^{th} channel is operating within calibration, otherwise the j^{th} channel's calibration is suspect. (3.22)

where:

$\bar{\mathbf{r}}_j$ is the mean residual between the parameter estimate and the training data for the j^{th} channel

σ_j is the standard deviation of the residual between the parameter estimate and the training data for the j^{th} channel

x_{kj} is the j^{th} element of the k^{th} observation vector not contained in the training data

p_k^f is the parameter estimate corresponding to x_{kj}

Detection of an out-of-calibration channel requires a method of determining when a given channel's residual exceeds some nominal value. Methods such as the Sequential Probability Ratio Test (SPRT) can be used to identify when a drift has occurred, but we will employ a much simpler, and more robust, method in this research. The chosen approach is to suspect an out-of-calibration alarm when a channel's residual falls outside of a $\pm 2\sigma$ band surrounding the residual mean value. A more practical criteria is to use

± 1.4% of the span to be consistent with the allowable drift [EPRI, 2000]. Both of these two values are used during this research and are specified accordingly.

3.4 Inferential Sensing

A portion of this research employs an inferential sensor to provide additional information during 2-channel monitoring. Principal Component Regression (PCR) was chosen as the inferential sensor model. Several techniques could be used, but since the relationships in the data set we studied have such high linear correlations, a linear technique such as PCR performs well. Principal Component Analysis is an unsupervised method that decorrelates the data. It attempts to reduce the dimensionality of the data while retaining the valuable information. PCR is a robust method commonly chosen to perform regression when the inputs are highly correlated. Simple linear regression is unsuitable for this type of problem due to the ill-conditioned nature of the fisher information matrix that is inverted during calculation of the pseudo inverse. For a more in depth discussion on regularization methods for inferential sensing see Hines [1999].

Let X be an nxm matrix with n observations and m variables. The data matrix X can be written as the product of an nxm column orthogonal matrix U, an mxm diagonal matrix S with positive or zero elements (the singular values) and the transpose of an mxm orthogonal matrix V:

$$X = U \cdot S \cdot V' \qquad (3.23)$$

The columns of V are the eigenvectors associated with the eigenvalues of X'X,which are the square root of singular values.Let us use the first k eigenvectors in V to define an mxk matix P=[v1,V2,…,vk]. Calculate the T scores using P, T=XP.

The principal component regression model is written as a linear combination of the score vectors.

$$\hat{y} = T \cdot \alpha \tag{3.24}$$

The inferential model designer has one major decision to make when developing a PCR model: which Principal Components to use in the regression model. Several methods exist, but we chose to retain only the first PC because it contains about 99% of the data set variance.

3.5 Merging Inferential Modeling with ICA

The hybrid Redundant Sensor Estimation Technique (RSET) is a combination of inferential sensing and Independent Component Analysis filtering. The method is illustrated by an example. Figure 3.1 presents a block diagram of the basic functional layout to predict first stage turbine pressure which is measured with two redundant sensors.

Several correlated signals are used to predict the turbine pressure using an inferential model. This prediction, along with the two redundant sensors, is processed by the ICA algorithm to produce an optimal estimate of the turbine pressure. Each of the sensor values is then compared to the estimate and residuals are formed. If the sensors are operating normally, the residuals should be fairly small random values near zero. If a sensor begins to drift, its corresponding residual will grow. A logic module is used to determine if the residuals are normal or come from faulty sensors. Several methods can be used including control charts or the more complex Sequential Probability Ratio Test developed by Wald [1947].

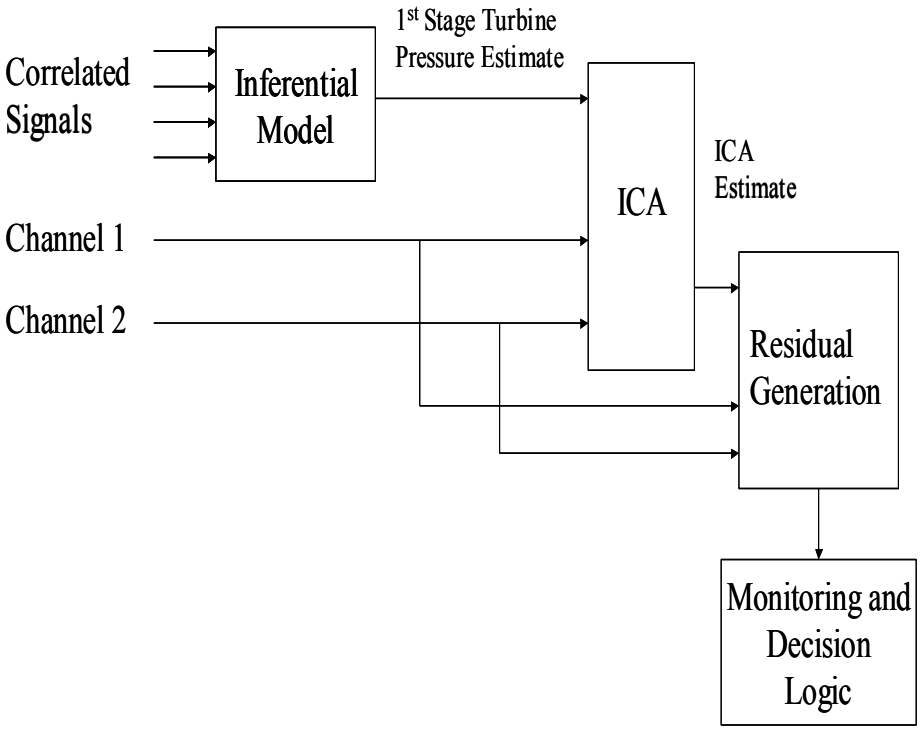

Figure 3.1. Hybrid inferential modeling and ICA block diagram.

Results of above scheme are discussed in Chapter 4. In Chapter 4, ICA is applied to both a simulated data set and real nuclear power plant data sets. The robustness and effectiveness of the ICA method, as well as applicable conditions, are shown by comparison with other methods, computer simulation, and a water tank level control experiment.

CHAPTER 4
RESULTS

In this chapter, a thorough study of ICA sensor validation method is presented. First, we begin with a simulated toy problem to resolve a signal from several independent noise sources. Second, real data sets from nuclear power plant are used for a comparison study between PCA, ICA and a simple average. Next, a drift detection example is presented and several characteristics such as variance reduction, robustness, batch mode window selection and selection rule are discussed.

In section 4.8, an ICA controller experiment reveals more features about the ICA method especially on non-stationary signals. The necessities for a unity check are presented and the unity check algorithm is integrated into the ICA controller experiment.

In section 4.9, a hybrid system merging inferential sensing and ICA is tested using the nuclear power plant data. Both a drift detection example and a robustness to input anomalies example of the new system are presented.

4.1 Simulated Data Sets

For the simulated data set, consider a source signal s given by the following equation:

$$s = 0.8 \cdot Sin(2\pi \cdot \sqrt{t}) + 0.2 \cdot t^2, t \in [0,3] \qquad (4.1)$$

The measured signal is a combination of the actual signal (*s*) and both common and independent noise sources. A common noise source (*nc)* is drawn from the laplacian

distribution (λ=0.7), and three independent noise sources are drawn from the normal distribution: $n_i \sim N(0,1)$. These noise sources are added to each channel with differing weights in order to model a contaminated signal measured by a process sensor. An example of a common noise source would be the high frequency boiling component of a signal measured by a steam generator level detector while an example of an independent noise source would be electrical noise contamination of the specific sensor signal. Three data sets (c1, c2 and c3) are constructed using the mixture models given below:

$$
\begin{aligned}
c1: &\begin{cases} x_1 = s + 0.1 \cdot nc + 0.1 \cdot n_1 \\ x_2 = s + 0.1 \cdot nc + 0.1 \cdot n_2 \\ x_3 = s + 0.1 \cdot nc + 0.1 \cdot n_3 \end{cases} \\
c2: &\begin{cases} x_1 = s + 0.3 \cdot nc + 0.1 \cdot n_1 \\ x_2 = s + 0.3 \cdot nc + 0.1 \cdot n_2 \\ x_3 = s + 0.3 \cdot nc + 0.1 \cdot n_3 \end{cases} \\
c3: &\begin{cases} x_1 = s + 0.01 \cdot nc + 0.3 \cdot n_1 \\ x_2 = s + 0.01 \cdot nc + 0.05 \cdot n_2 \\ x_3 = s + 0.01 \cdot nc + 0.05 \cdot n_3 \end{cases}
\end{aligned}
\qquad (4.2)
$$

Set $c1$ represents fairly equal amounts of common and independent noise, set $c2$ represents a larger common noise contribution, and set $c3$ represents unequal amounts of independent noise with a minimal amount of common noise. Figure 4.1 shows the three signals of $c3$.

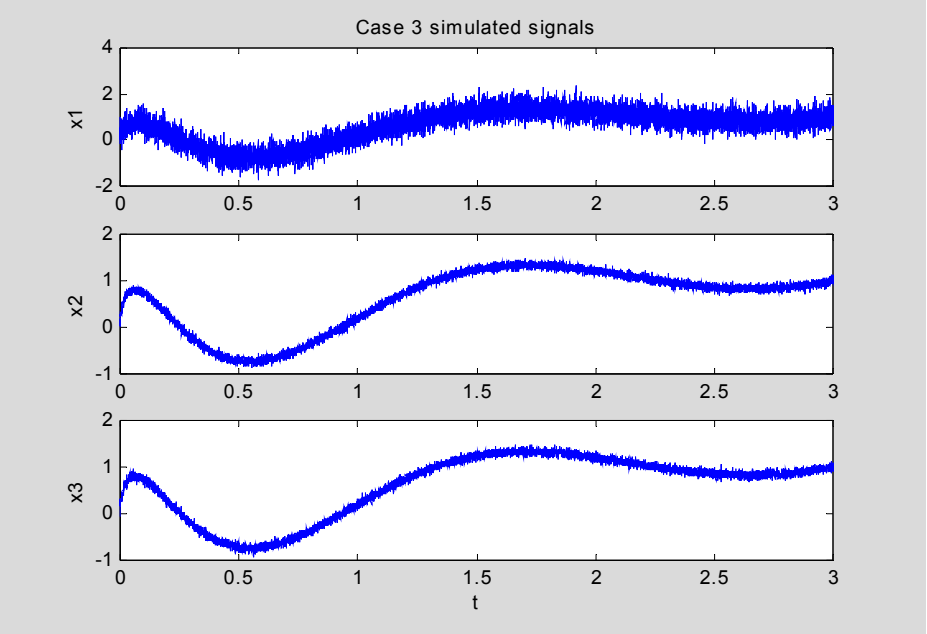

Figure 4.1. Mixed signals of $c3$.

The first step of the ICA algorithm is to decorrelate, or whiten, the signals. This step is equivalent to performing PCA. The principal components of set $c3$ are shown in Figure 4.2. Note that the signal of interest is contained in the top plot, which is the first or primary principal component, while the bottom two plots contain mixtures of the noise sources.

Next, the ICA algorithm (FastICA) [Hyvarinen, 1999] is applied to separate the independent components. Figure 4.3 presents the results of applying the ICA algorithm to set $c3$. Note that the signal is contained in the top plot and the noise sources are in the bottom two plots. Also note that the signal have less noise contamination when compared to the PCA results shown in Figure 4.2.

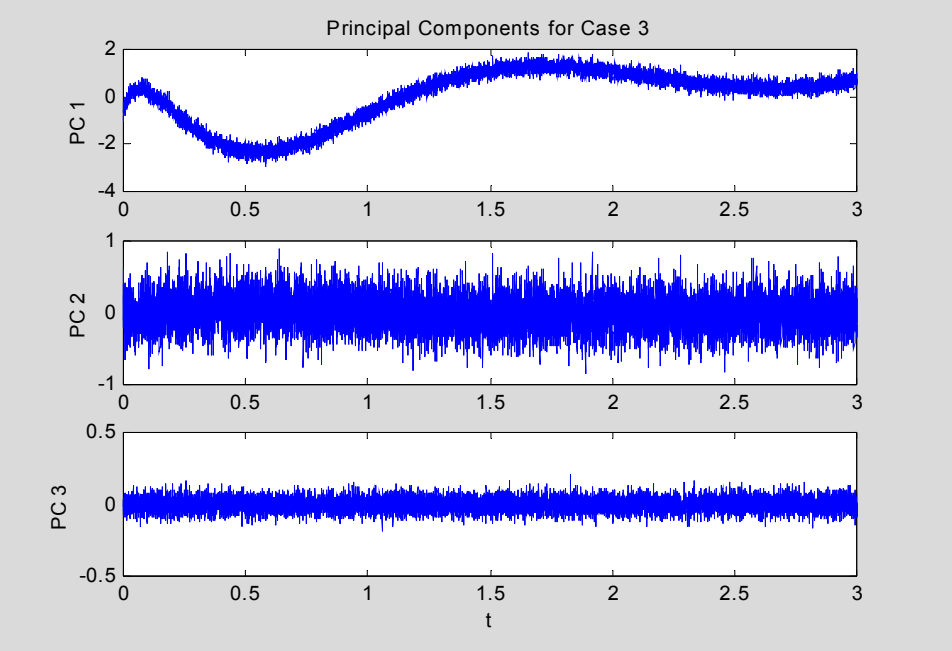

Figure 4.2. Whitened signals (principle components) of $c3$.

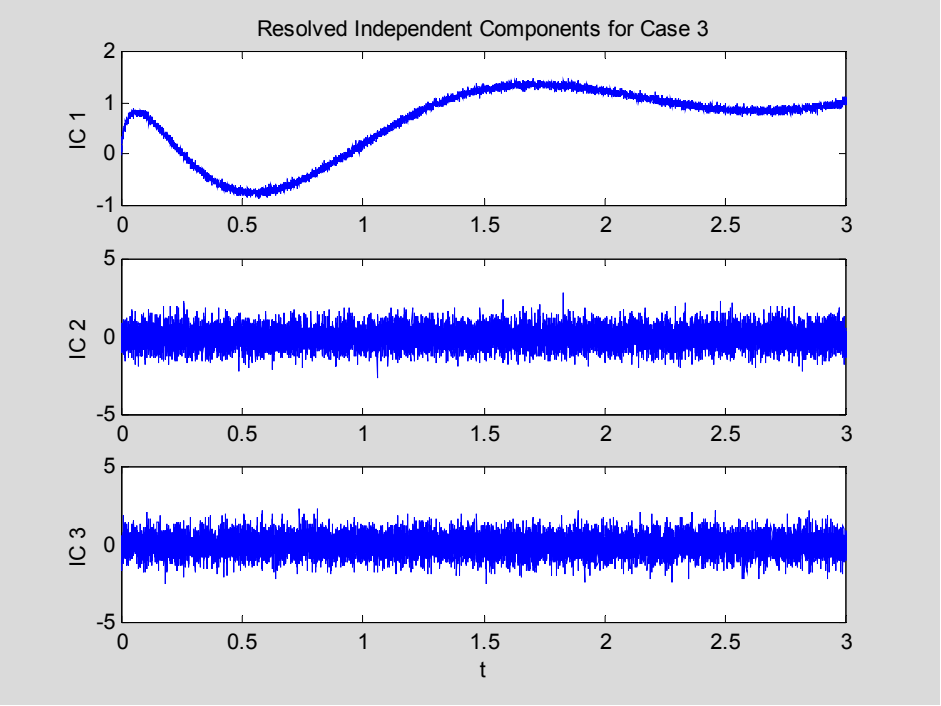

Figure 4.3. Resolved independent components.

4.2 Real Data Sets

Three sets of redundant channel data were obtained from a nuclear power plant at a sampling rate of 15 minutes. Each data set contains approximately 37,000 samples covering operations from March 2001 to April 2002. Five pressurizer pressure channels comprise the first redundant set, three pressurizer level channels comprise the second set, and three steam generator pressure channels comprise the third (see Figure 4.4).
The mean and standard deviations for the ICA, PCA, and Simple Average (SA) parameter estimates are shown in Table 4.1. Many ICA algorithms from the freeware ICA Toolbox [Cichocki, 2002] were used and show consistent results. The results from two of the algorithms are presented.

The results show that the ICA algorithm produces parameter estimates with standard deviations similar to those obtained via simple averaging. Additionally, standard deviations are consistently higher for the PCA parameter estimates. These results are consistent with those obtained for the simulated data sets. The mean values of the parameter estimates are shown to indicate that the parameter estimates for all of the methods are varying at or around similar mean values. The lesser varying ICA parameter estimates indicate a lesser uncertainty surrounding the estimations, and thus a more accurate result. Table 4.2 presents the standard deviations associated with the residuals computed for each individual channel, based on the corresponding parameter estimate, for each method. The standard deviations shown in table 4.2 were computed from the residuals between the parameter estimate for the given data set and each of the individual channels of that set. Changes in these residual signals are analyzed to determine the calibration status of the monitored channels.

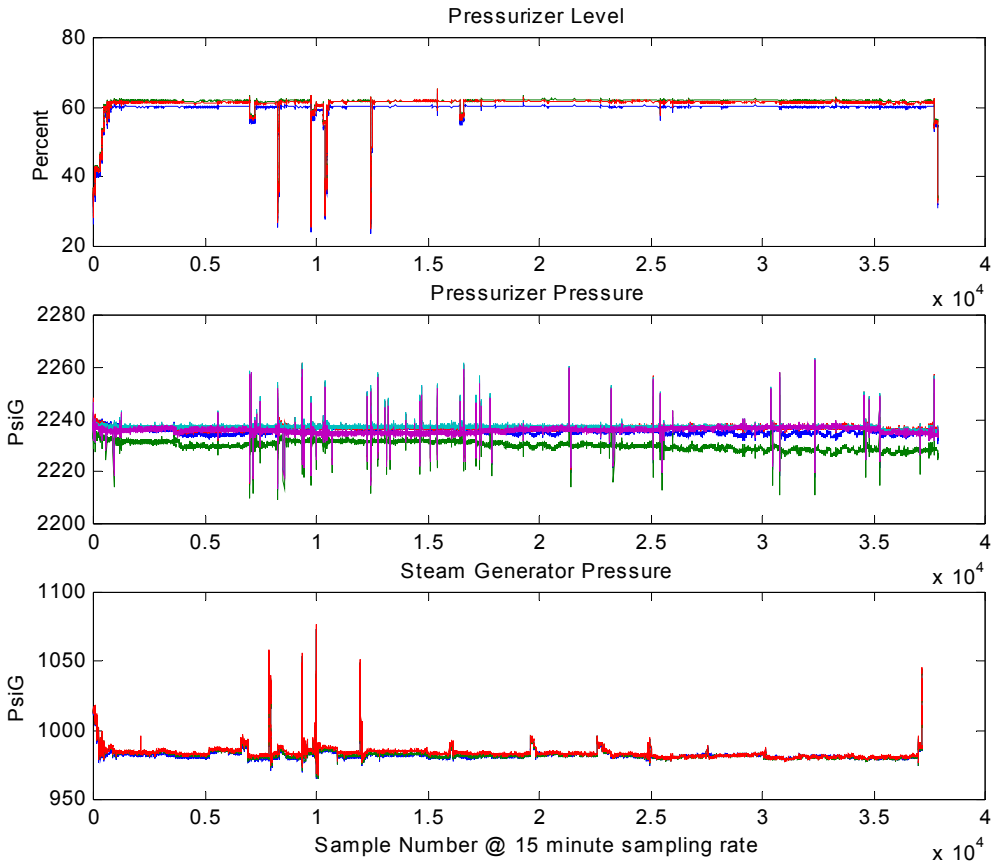

Figure 4.4. Redundant channel data.

Table 4.1 Parameter estimation results for redundant channel data

Estimation Method	Pressurize Level		Pressurize Pressure		SG Pressure	
	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation
Simple Average	60.7295	2.9635	2235.04	1.05	982.81	4.3645
ICA_JADEop	60.6084	3.3952	2236.28	0.963	983.17	4.2625
ICA_FPICA	60.5254	3.2351	2236.23	0.953	983.21	4.234
PCA	60.7295	5.1330	2234.31	2.45	982.83	7.56

Table 4.2 Redundant channel parameter estimation residual standard deviations

	Channel	ICA_FPICA	PCA	SA
Pressurizer Level (%)	1	0.3966	2.1707	0.0716
	2	0.5104	2.1722	0.1090
	3	0.4775	2.1715	0.0958
Pressurizer Pressure (PsiG)	1	0.9906	1.2135	0.6459
	2	1.4434	1.1127	1.0812
	3	0.5015	1.6664	0.4520
	4	0.1760	1.7097	0.4203
	5	0.8088	2.2376	1.0482
Steam Generator Pressure (PsiG)	1	0.3387	3.4133	0.6187
	2	1.0292	3.1869	0.2907
	3	1.3662	3.1213	0.6321

The results indicate that the PCA residuals are consistently higher than the ICA residuals, while both are generally higher than the residuals determined when using the simple average of the channels as the parameter estimate. It is important to note that it is not significant that the SA residual standard deviations are slightly lower than the ICA standard deviations. The reason the residual standard deviations for the simple average are noticeably smaller, in most cases, is that the SA parameter estimate is more consistently centered within the data of the redundant channels due to the nature of the averaging process. The slight increase in the standard deviations incurred for the ICA parameter estimates are not excessively large. More importantly, the ICA estimation process is less influenced by individual channel variations than the simple average. This more important feature is an indispensable attribute of a redundant channel monitoring approach because it must be robust to individual channel trends in order to realize out-of-calibration conditions.

When designing a sensor calibration monitoring system, one of the most important considerations is the influence of the noise or drift component of an individual signal on the overall model. In the case of the simple average, the influence is obvious. Under conditions when an observation occurs at a relatively large distance from the nominal process value (spike), each model behaves differently. There are two different scenarios to consider: when the deviation exists in only one of the set of redundant channels (unique spike), and when the deviation exists in all channels (common spike). Under the unique spike scenario, the ICA parameter estimate residuals are modestly influenced, whereas the PCA parameter estimate residuals are more strongly influenced. Under the common spike scenario, the ICA parameter estimate residual shows no evidence of the occurrence due to the estimation closely matching the deviant channel measurements. If all of the channels are reporting the same large deviation, ruling out common-mode failure, it is assumed that the measurements are accurately reporting the process conditions. For PCA, under the common spike scenario, the parameter estimate over-emphasizes the deviation due to the variance maximization basis of the model. Thus, the PCA parameter estimate residuals show obvious evidence of the occurrence. Under both spike scenarios, the PCA parameter estimate residuals are more strongly affected and thus will exhibit a greater standard deviation in the parameter estimate residuals. Noting that the standard deviations from the simple average parameter estimate residuals are lower, one might consider this technique. However, due to the mutual dependence of the parameter estimate on each channel, the resultant influence of the drifting channel on the parameter estimate is much more pronounced when simple averaging is employed rather than ICA. The drift identification example in the next

section gives evidence of the ICA model's ability to limit the influence of a drifting signal on the overall parameter estimate, thus leading to a more rapid identification of drift and subsequent out-of-calibration annunciation.

4.3 Drift Detection

The redundant channel data for this case study was acquired from an highly redundant nuclear power plant. The original data set had nine redundant channel measurements with one experiencing an actual channel drift during operation. As the number of redundant sensors is increased, the spillover in conventional techniques decreases and the drift detection problem becomes easier. For this exercise, we select only three of the original nine redundant signals to simulate a U.S. system made up of three redundant channels. The drifting channel is included as channel #2.

An ICA transform was carried out on the selected window which range from 0 to 800. The drift channel was included during model construction. Figure 4.5 shows the original three sensor measurements and the ICA parameter estimate. From a visual inspection, the ICA estimate contains no drift. The next three figures (Figure 4.6) show the residuals of three of the sensors with their drift detection error bands.

The error bands for each channel are determined from equation (3.22). The variances are estimated from first 100 data points. Figure 4.6 shows the drift detection results using the sensor residuals and calculated error bands. Channel 2 is clearly identified as a drifting channel with channels 1 and 3 remaining within their error bands. In actual application, the error bounds may be significantly larger because the sensor is allowed to drift before the calibration bounds are violated.

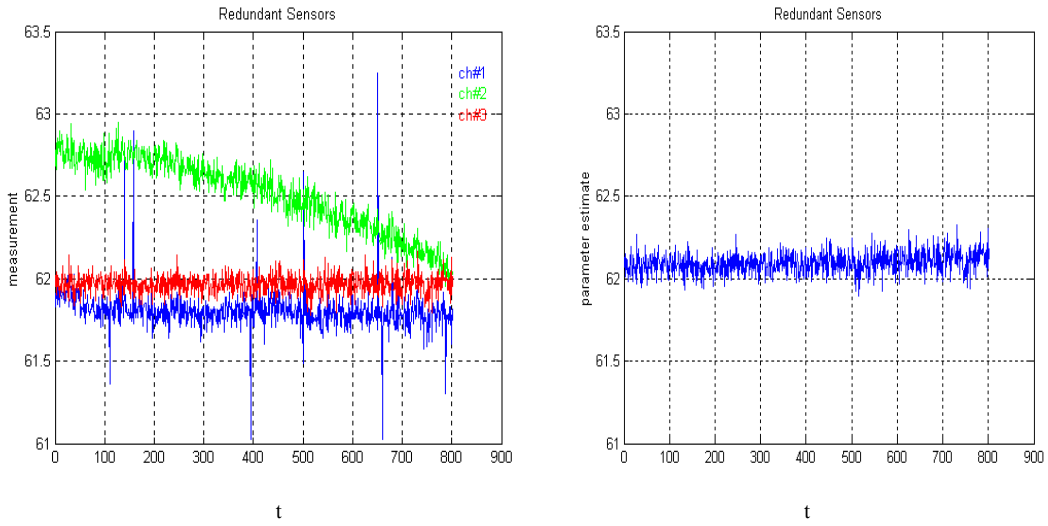

Figure 4.5. Redundant channel measurement and ICA estimate.

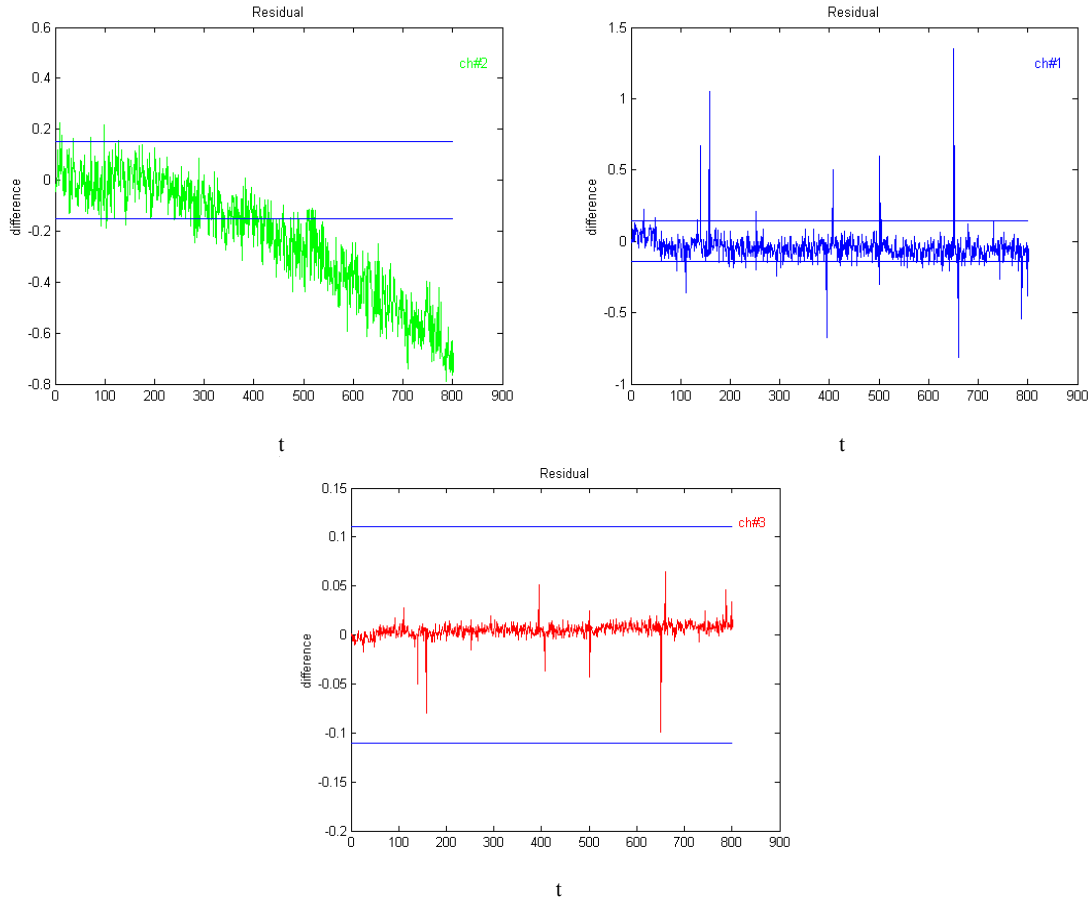

Figure 4.6. Drift detection from predicted residual and error band.

In addition to the ICA results, ICMP results have been performed for comparison and are included as Figure 4.7. The ICMP algorithm identifies channel 2 as a drifting channel. Unfortunately, it also identifies channel 3 as a drifting channel. The reason for the incorrect identification of the drifted channel is that the ICMP estimate is drifting due to spillover. The ICA method has the advantage of being resistant to spillover, thus making it a robust technique.

Next, we construct a more difficult case. The same nuclear power plant data is used but only two redundant measurements are retained, one of which is the drifting channel. Figure 4.8 shows the original sensor signals, the parameter estimate, and the two residuals with their drift bands. Since ICA parameter estimate is accurate and drift free, residual plots for channel 1 and channel 2 show the drift components in each channel. The drift channel (Ch#1) is clearly identified.

4.4 Variance Reduction

The second application is to use ICA as a pre-processor to estimate the process parameter more accurately. The redundant channel data is five pressurize pressure channels obtained from a nuclear power plant at a sampling rate of 15 minutes. The data set contains approximately 37,000 samples covering operations from March 2001 to April 2002. Figure 4.9 shows the data set and ICA parameter estimate.

In this case, ICA estimate contains less noise than simple average and even each individual channel. The results are shown in Table 4.3. ICA estimate weighted noisy channel less while simple average equally weighted each channel. The non-Gaussian noise has been filtered by ICA; thus, ICA is ideal as a pre-processor to filter noise.

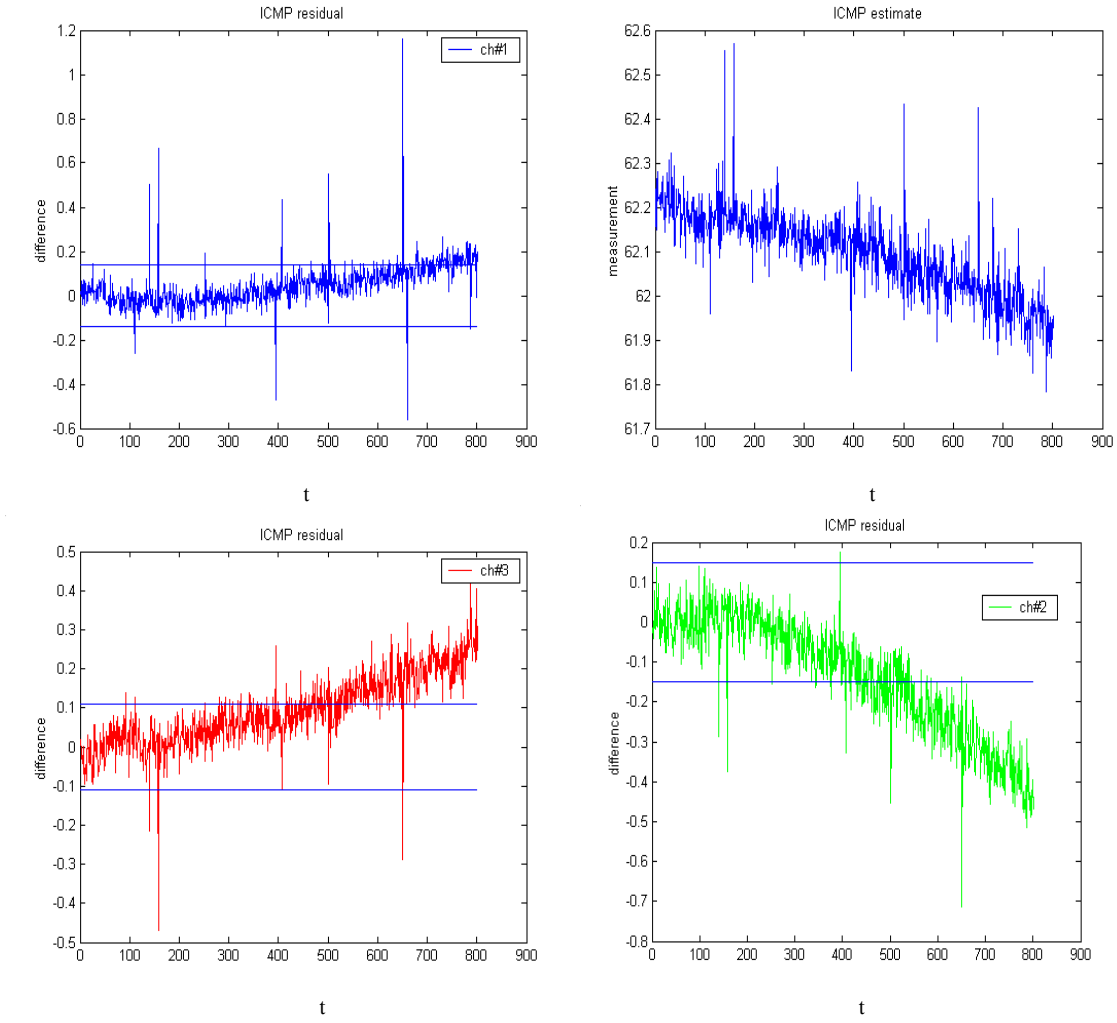

Figure 4.7. ICMP results for drift detection.

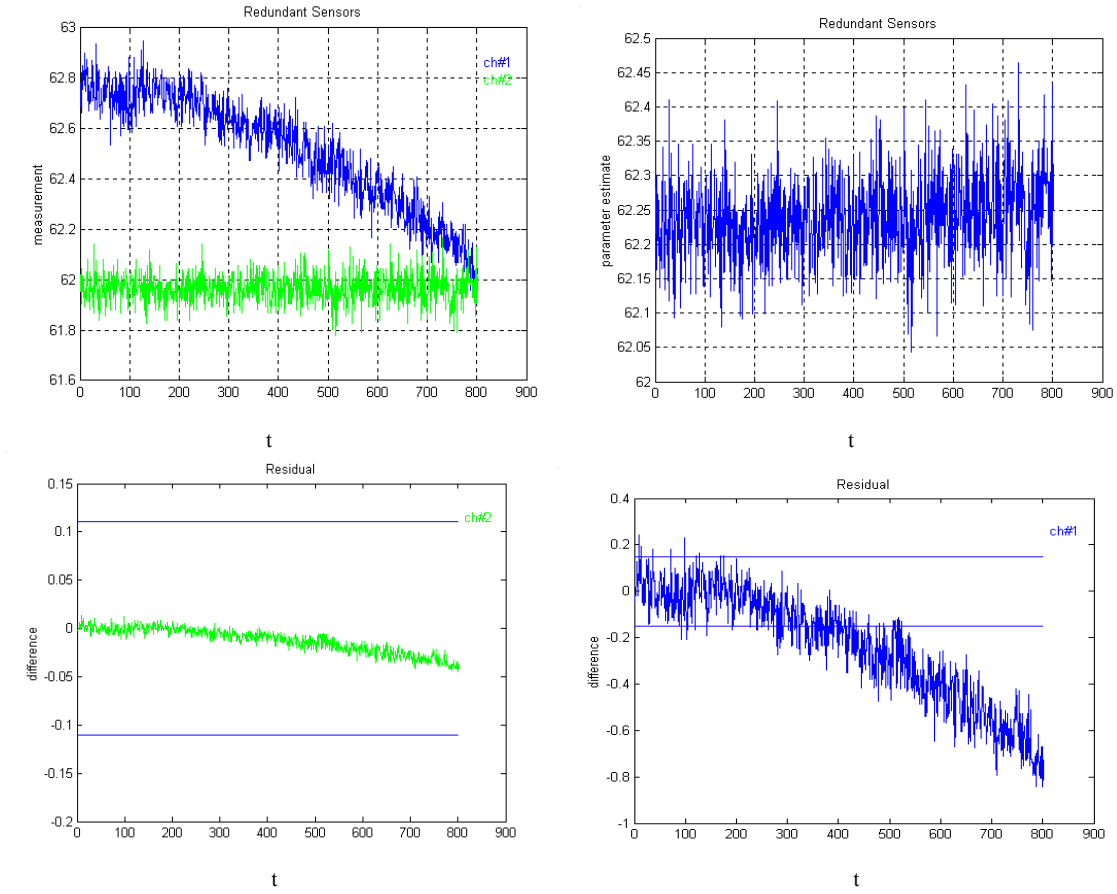

Figure 4.8. Two redundant channel case and drift detection.

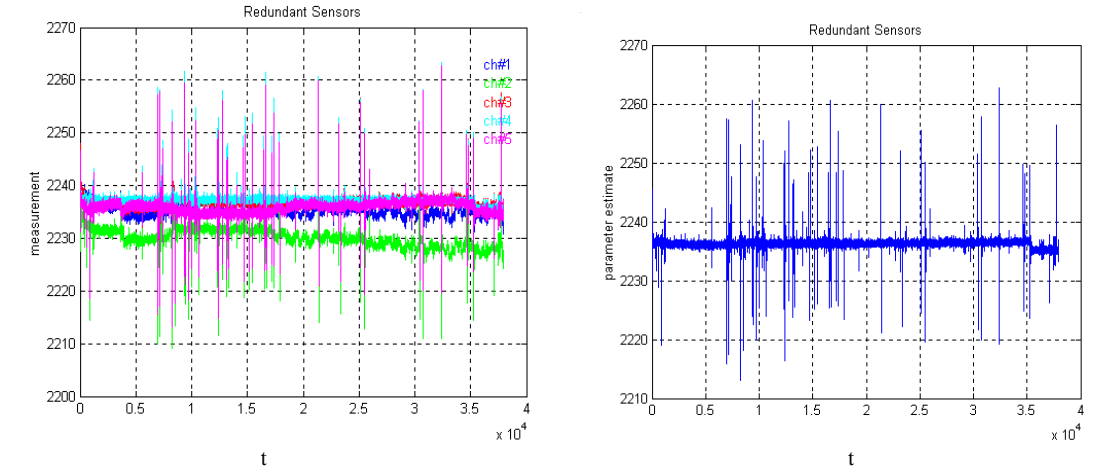

Figure 4.9. Pressurizer pressure measurements and the ICA-based parameter estimate.

Table 4.3 Compare variance of ICA prediction and individual channels

	Variance
Channel 1	1.8915
Channel 2	3.0477
Channel 3	1.1434
Channel 4	0.9522
Channel 5	1.5604
Simple Average	1.1058
ICA Prediction	0.8796

4.5 Robust Testing

These results show the ICA applicability for drift detection and variance reduction. In order to test the robustness of the method, bootstrap sampling is used. Bootstrap is a method, related to Monte Carlo analysis, that applies the technique many times and analyzes the variability of the predictions. This section presents experimental evidence to quantify the robustness of the ICA method.

First, a drift case was artificially created. A linearly increasing drift of 2% is inserted into the first channel of the five-channel pressurize pressure data set beginning at time stamp 15000. This drift is a linear increase in the pressurizer sensor value. Figure 4.10 shows the simulated result.

Next, a 1000 sample bootstrap experiment is simulated. For each bootstrap sample, a slightly different observation set is chosen. An ICA transform vector is calculated for each sample and the resulting ICA weights are shown in figure 4.11. The statistics of 1000 ICA vectors are shown in Table 4.4. The standard deviations shown in the table are less than 1% of the weights.

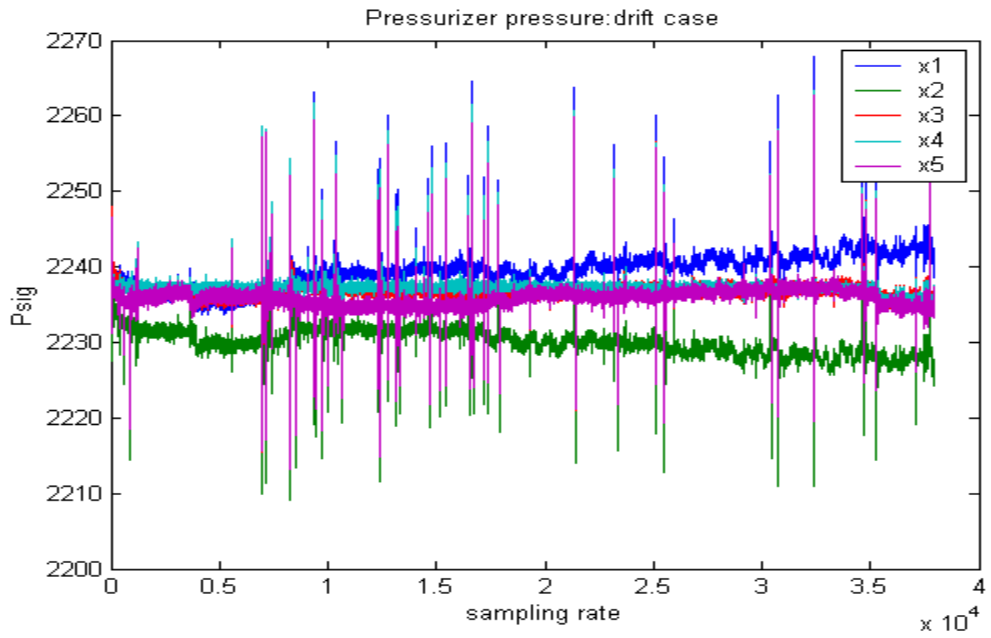

Figure 4.10. A simulated drift case in the redundant measurement.

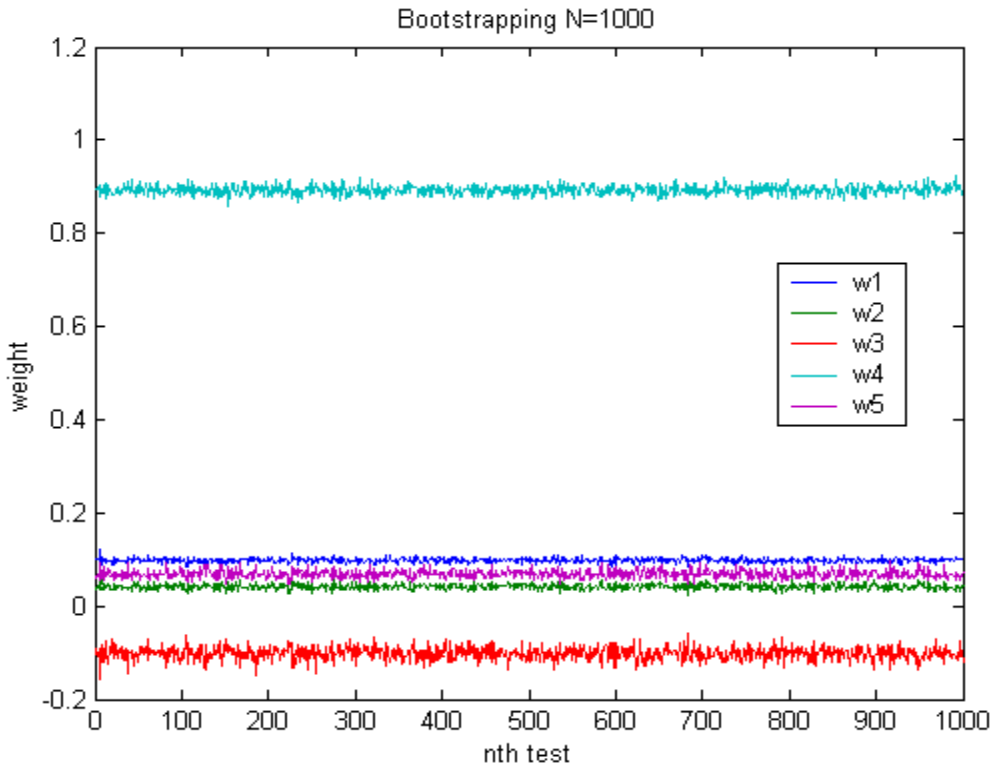

Figure 4.11. ICA weights from bootstrap results.

Table 4.4: Statistics of ICA weights from 1000 bootstrap

	Mean	Standard Deviation
w_1	0.0983	0.0043
w_2	0.0414	0.0056
w_3	-0.1016	0.0136
w_4	0.8928	0.0093
w_5	0.0686	0.0084

This sensitive test shows the ICA transform vector changes little for the little change of the training data. This means the technique is stable and repeatable.

4.6 Batch Mode Processing

On-line sensor calibration monitoring is not required to be performed in real time. "on-line" is referred in the sense that the monitoring is performed while the process is operating. There is always a certain time delay for the calibration monitoring system to determine the sensor status. The reason is two fold: 1. for some sensor fault types, i.e. a slow drift, it takes a certain amount of time for the drift to become statistically significant and be detected, and 2. it always takes a finite amount of data to build the model. Generally speaking, as more data is incorporated into the model, the model becomes more accurate. Therefore, the time resolution for on-line monitoring cannot be instantaneous. One can refer to this as batch mode processing.

Two types of batch mode processing are studied. One is to fix the starting point of the training data and increase it as new data are acquired. The second is to fix the window size, and move the window at each step. Both of these methods are examined in the following example.

The two-redundant sensor case from a nuclear power plant data in Figure 4.8 is given below for illustration. The first channel contains a drift. Since the parameter estimate is a linear combination of the first channel measurement and the second channel measurement, for *drift detection* purposes, one can simply look at the weight coefficients of the corresponding channels. As the drift occurs, the first channel weight becomes smaller, signifying that channel is drifting and its information is not included in the prediction.

We use 100 data points as the window size. Tables 4.5 and 4.6 contain the model coefficients. The first table contains coefficients corresponding to the first sensor while the second table corresponds to the second. If both sensors are operating properly, the coefficients should equally weigh the sensors with coefficients equal to 0.5.

The rows of the table correspond to the starting point of the window and the columns correspond to time in increments of 100 points. For the first row, the starting point is 1 and each column increments the window by 100 points. The model corresponding to the first row and the second column of the coefficient matrix, the training data contains 200 points. For the second row and the second column data, the starting point is 101, and so on. The matrix diagonal is simply a moving window with the window size of 100. The results are shown below with very small coefficients corresponding to drift detections highlighted.

This exercise shows that an anchored beginning point is a better method. Note that the first row detects the drift in the 4^{th} column corresponding to 400 data points. The moving window does not detect the drift until time step 500 and thereafter, the drift detection is uncertain as shown by the poor prediction at the 700 time step.

Table 4.5. Channel 1 ICA weight coefficients

Start with	Window size	100	200	300	400	500	600	700	800
1		0.4622	0.5338	0.3274	-0.0926	0.0613	-0.0589	-0.0090	0.0048
100		0	0.6833	0.4511	0.0372	0.1323	-0.0322	0.0017	0.0212
200		0	0	0.8400	0.6661	0.1843	-0.1358	-0.0008	-0.0260
300		0	0	0	0.4725	0.0304	0.1153	-0.0034	0.0606
400		0	0	0	0	0.1414	0.2464	0.2534	0.1016
500		0	0	0	0	0	0.1756	0.3822	0.5567
600		0	0	0	0	0	0	0.4667	0.5592
700		0	0	0	0	0	0	0	0.2445

Table 4.6. Channel 2 ICA weight coefficients

Start with	Window size	100	200	300	400	500	600	700	800
1		0.5381	0.4657	0.6747	1.0995	0.9436	1.0647	1.0140	0.9996
100		0	0.3144	0.5494	0.9681	0.8717	1.0375	1.0029	0.9828
200		0	0	0.1560	0.3321	0.8190	1.1419	1.0051	1.0301
300		0	0	0	0.5278	0.9741	0.8881	1.0073	0.9425
400		0	0	0	0	0.8618	0.7557	0.7484	0.9008
500		0	0	0	0	0	0.8266	0.6185	0.4429
600		0	0	0	0	0	0	0.5334	0.4404
700		0	0	0	0	0	0	0	0.7562

The difficulty in implementing this method is to know when to anchor the window starting point. This requires detecting the drift, then anchoring the window starting point. One solution is to apply different detecting sensitivities on drift detection and window size selection by setting a narrower band for the first purpose.

4.7 Selection Rule

Another feature of ICA modeling is termed the "selection rule". This rule states that the correlation coefficient between the ICA estimate and the individual measurement versus the measurement variance holds a linear relationship. A lower measurement variance produces a higher correlation coefficient. Thus, ICA selects the lowest variance channel as the most heavily weighted channel. This is illustrated in the following example.

The pressurizer pressure data set again is used to illustrate this example. The data is plotted in Figure 4.12. A linear regression line between these correlation coefficients is plotted in Figure 4.13. The standard deviation and mean of each channel and the correlation coefficients with the ICA prediction are listed in Table 4.7. One should notice that the correlation coefficient for the first channel is about 0.69. In the next example this channel will be artificially drifted.

Next, the drift case is tested. A linearly increasing drift of 2% is inserted into the first channel of the same data set beginning at sample 15000. It is plotted in Figure 4.14. A new ICA parameter estimate is calculated from the drift case.

The correlation coefficient for the drift channel (ch#1) goes down to 0.39. A linear relationship still holds and it is plotted in Figure 4.15. Table 4.8 shows the

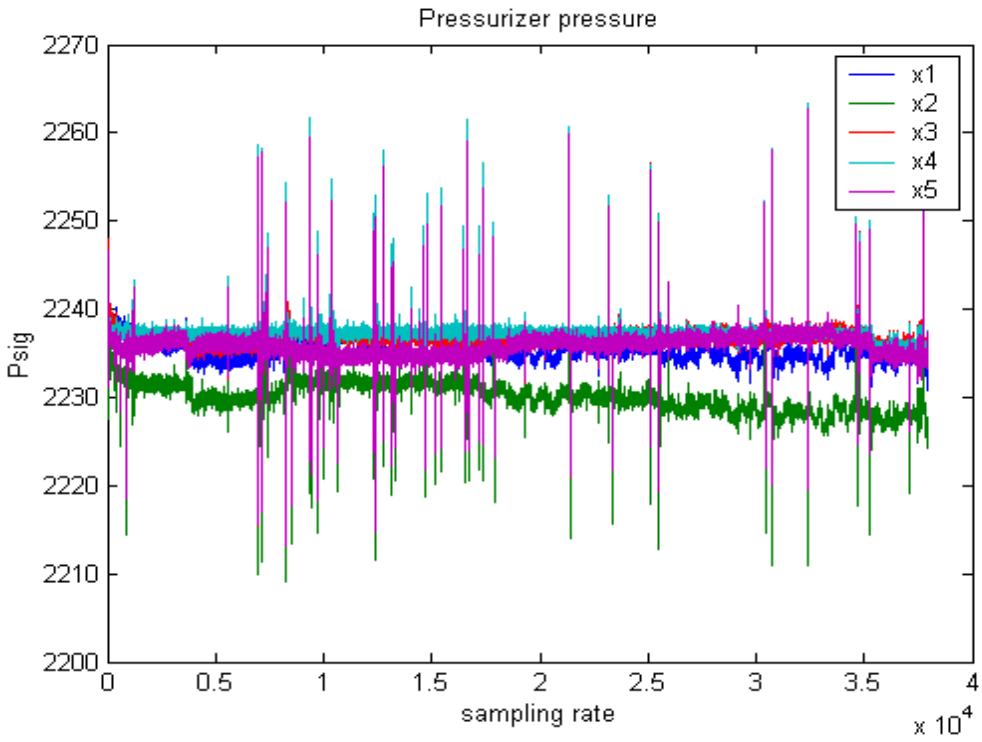

Figure 4.12. Pressurizer pressure data set.

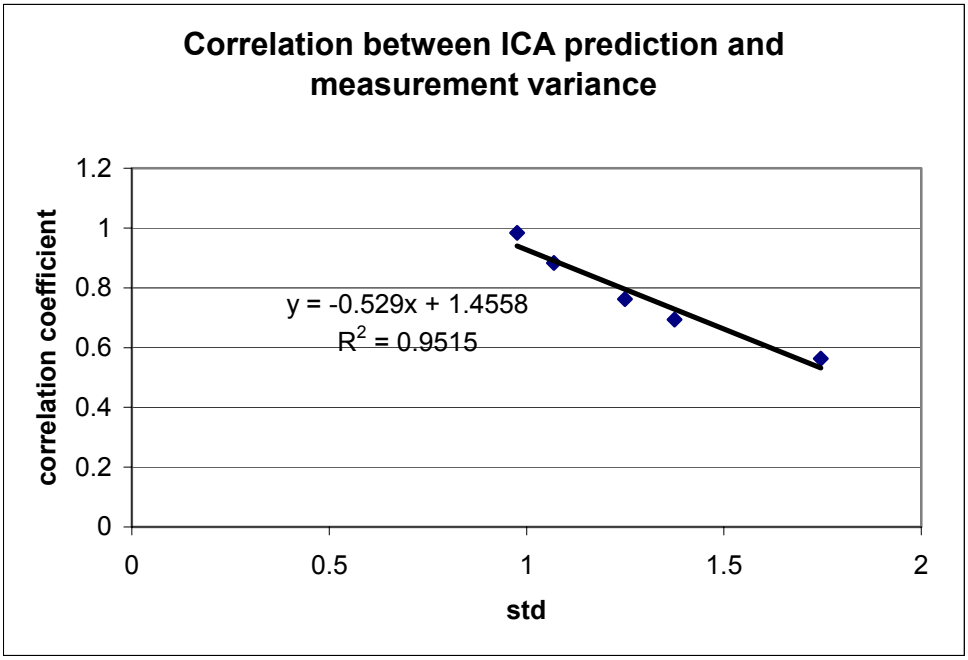

Figure 4.13. Correlation between ICA prediction and measurement variance.

Table 4.7 Statistics of pressurizer pressure data

	Mean	Standard Deviation	Correlation with ICA Prediction
Channel 1	2235.5	1.3753	0.6937
Channel 2	2230.0	1.7458	0.5628
Channel 3	2236.6	1.0693	0.8833
Channel 4	2237.2	0.9758	0.9836
Channel 5	2235.9	1.2491	0.7621

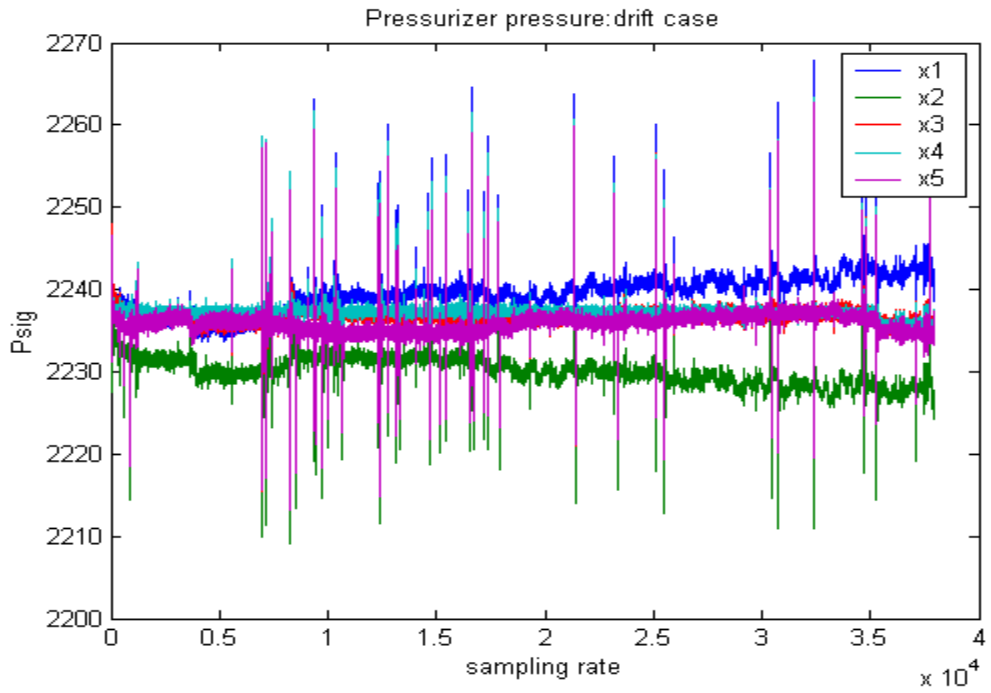

Figure 4.14. Drift case for pressurizer pressure.

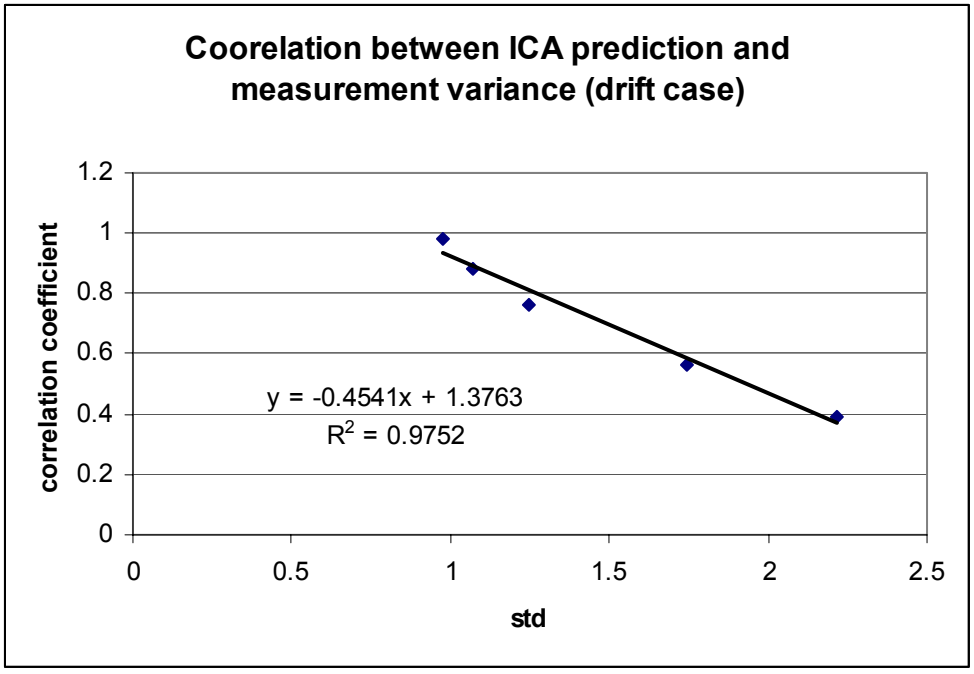

Figure 4.15. Correlation between ICA prediction and measurement variance (drift case).

Table 4.8 Statistics of pressurizer pressure data (drift)

	Mean	Standard Deviation	Correlation with ICA Prediction
Channel 1	2239.4	2.2194	0.3912
Channel 2	2230.0	1.7458	0.5643
Channel 3	2236.6	1.0693	0.8842
Channel 4	2237.2	0.9758	0.9835
Channel 5	2235.9	1.2491	0.7616

calculation results. The selection rule reveals the fundamental reason that why the ICA model works well for both drift detection and variance reduction. The ICA estimate weighs the higher variance channel less giving it less importance in the final parameter estimate.

4.8 ICA Controller

A water level measurement and control system using ICA was developed for real-time experimental studies. The experimental setup up was a very important portion of this research because it provided the ability to examine several different application line-ups. In previous sections, actual reactor data was examined. Although one would assume this type of data to be optimal in this study, it is very limited in that nuclear power plants usually operate at steady state conditions. To assure the methodology is robust, it must be tested in all condition in which it may be applied. The conditions consist of the number of redundant sensors plus whether the signals are steady state (stationary) or transient (non-stationary). The setup has three sensors in which two or three can be used.

Additionally, the state can be steady state with use of a controller, or transient if manually controlled as such.

Second, when the sensor is used for control purpose, the system will be drifting if a drift sensor is selected as the controller input. Conventional FDI scheme cannot identify the drift sensor under this condition. The use of ICA estimate as controller inputs solved this problem and it is tested in the experiment.

4.8.1 Experiment Setup

The experimental setup consists of a water tank, three Rosemount differential pressure transducers, an air-operated control valve, a National Instrument DAQ system and a laptop computer. Three differential pressure sensors [Rosemount 3095, 3051] are used for redundant level measurement. One of the sensors has an RTD for mass flow rate temperature compensation and is used to inject simulated drifts by heating the RTD during measurements. A NI LabView 7.0 program was developed for collecting the real-time data collection, analysis, and control. The basic system block diagram is shown in Figure 4.16.

Since the system to be controlled is non-linear (flow is a non-linear function of water level), a standard PID controller performs poorly; therefore, a fuzzy logic based controller was developed that used rules based on level error. This resulted in better response characteristics, which allowed for steady state operating characteristics at all water levels. A picture of the experimental setup is provided as Figure 4.17.

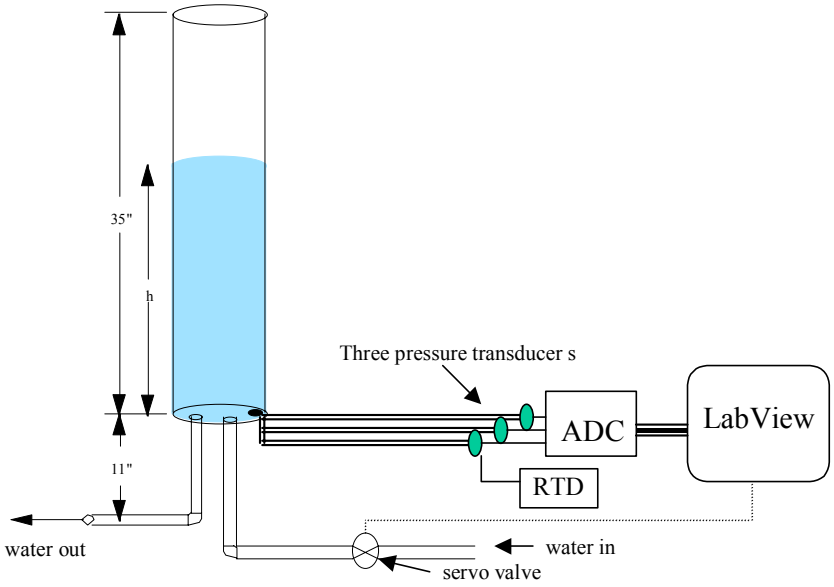

Figure 4.16. Experimental setup block diagram.

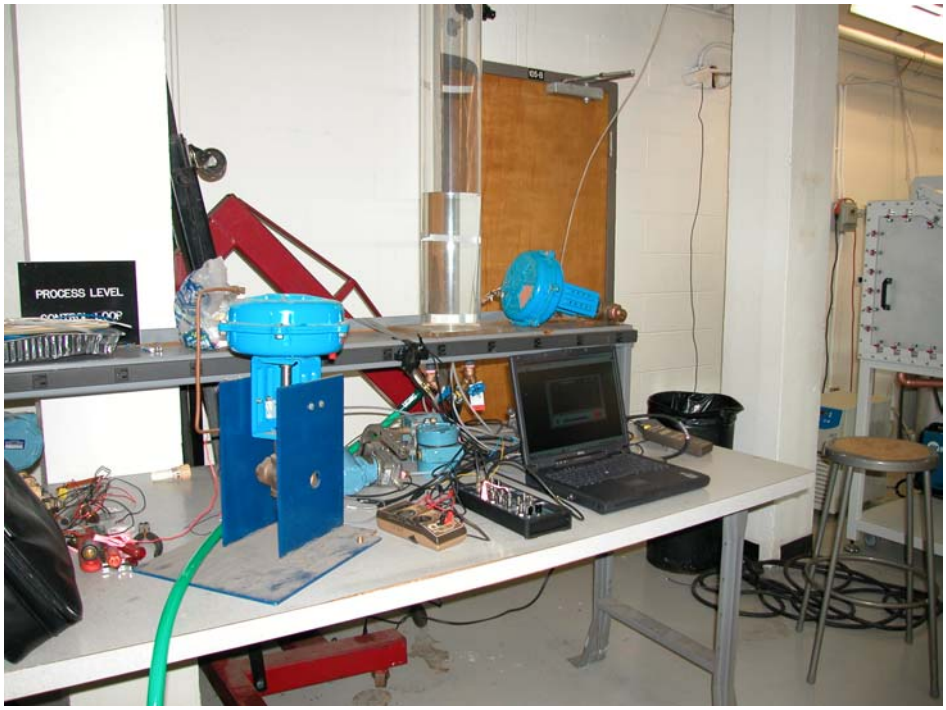

Figure 4.17. Experiment setup picture.

This section presents the results of applying the ICA algorithm to several experiments in real time. The results presented in the figure use a different method of displaying the fault condition. In the Figure 4.18, a measurement uncertainty is roughly estimated from residual variance of the first 100 points. It is shown as a band around the residual, and the re-calibration set point is shown as control tolerances. When the uncertainty band crosses the tolerance set point, the sensor is determined to be a faulty sensor.

4.8.2 Steady State Measurements

The first test was developed for the test of steady state response characteristics. Both the inlet and outlet valves were opened and the water level was allowed to reach a

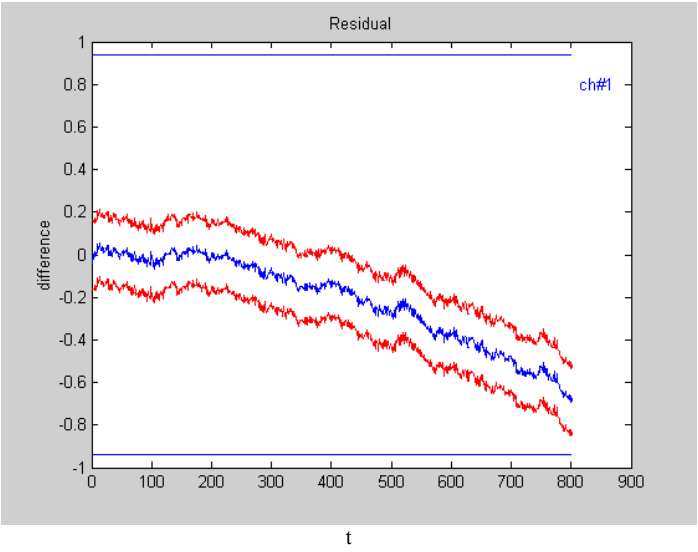

Figure 4.18. Tolerance bands incorporating residual uncertainty.

steady state condition. Two fault types were then injected into one sensor: a step change fault and a slow drift fault. The step change fault was injected by changing the position of one sensor suddenly while maintaining water level. The slow drift fault is injected by heating the RTD temperature compensation.

Figure 4.19, presents the results for the step change case using three redundant sensors. The model covers data from three sensors when a step change is injected into channel 1. The first plot shows the three sensor signals. Note that the sensors are not precisely calibrated at the beginning of the experiment but are considered to be correct. The next three plots show the three channel ICA residuals with 3σ error bands. Channel one is easily determined to be the faulty channel. Note the spillover into the other channels is very small. The tolerance bands were set at 1.5%.

Figure 4.20 presents the results of an experiment using only two redundant sensors. The drift is injected into channel 2 by heating RTD with the total drift being about 1%. Again the first plot is the two sensor signals operating at steady state. The next two plots are the residuals of the two sensors. The control charts easily indicate that channel 2 is the drifting channel. Comparing with ICA, simple average residuals contain spillover and drift channel cannot be isolated. The result is shown in the last plot of Figure 4.20.

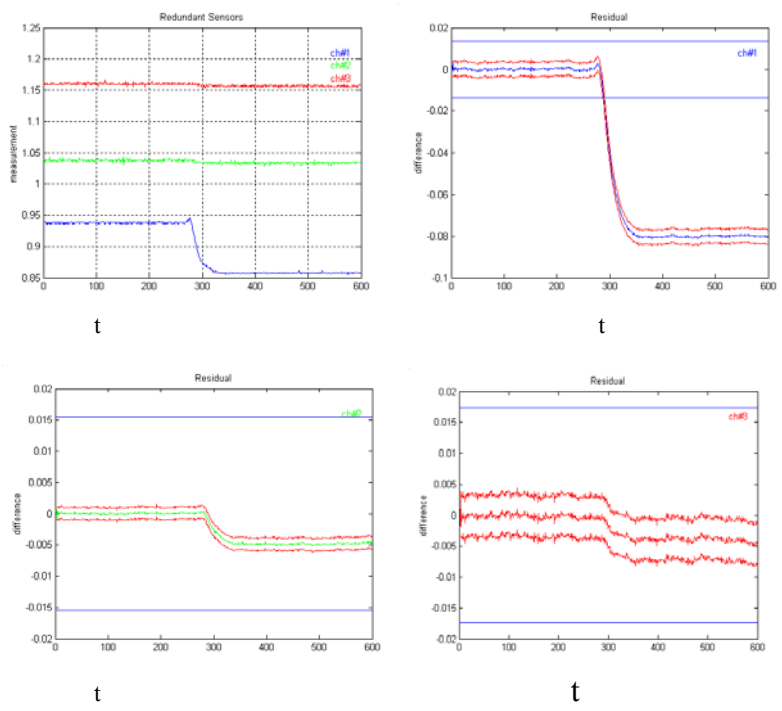

Figure 4.19. Step change fault detection.

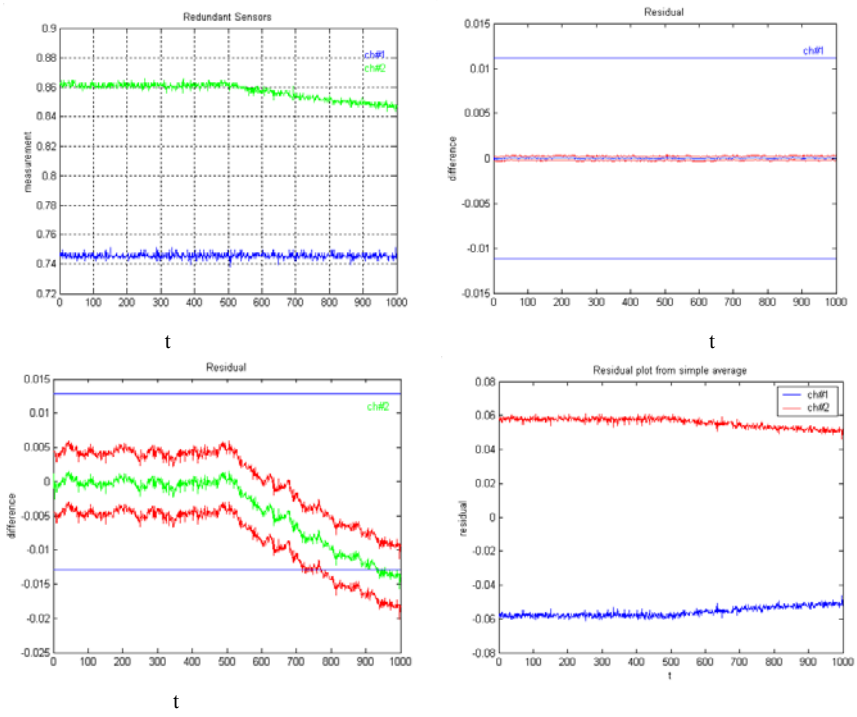

Figure 4.20. Slow drift fault detection of two sensors: ICA and simple average.

4.8.3 Non-stationary Measurements

The ICA algorithm was originally developed for stationary signals. It is a method to separate mixtures of independent signals and uses a method to maximize the non-Gaussian nature of the signals. Recall that when signals are mixed, the result is more Gaussian than the original signals, so one method of unmixing may be to find an unmixing matrix that maximizes the non-Gaussian nature. When we discuss the type of distribution being non-Gaussian, we are assuming the statistical nature is stationary. This is more stringent than specifying the type of noise as being Gaussian or having other characteristics that are stationary. In actuality, no process signal is going to be perfectly stationary, but under what conditions can the ICA based technique be used? In this section we test its application to non-stationary signals with different degrees of redundancy. We also test methods to transform non-stationary signals into stationary signals.

When originally testing the system on non-stationary measurements, we determined that the ICA algorithm did not perform correctly and the sum of the ICA coefficient is not equal to unity. It was also determined that this non-unity can be used to detect when the ICA system is operating outside its assumption of stationarity. Normally, before the process achieves steady state and during the time it changes from one state to another, the measurements are non-stationary. We will first examine a transformation method to make the signals more stationary. An example of a non-stationary measurement and the residuals are shown in Figure 4.21.

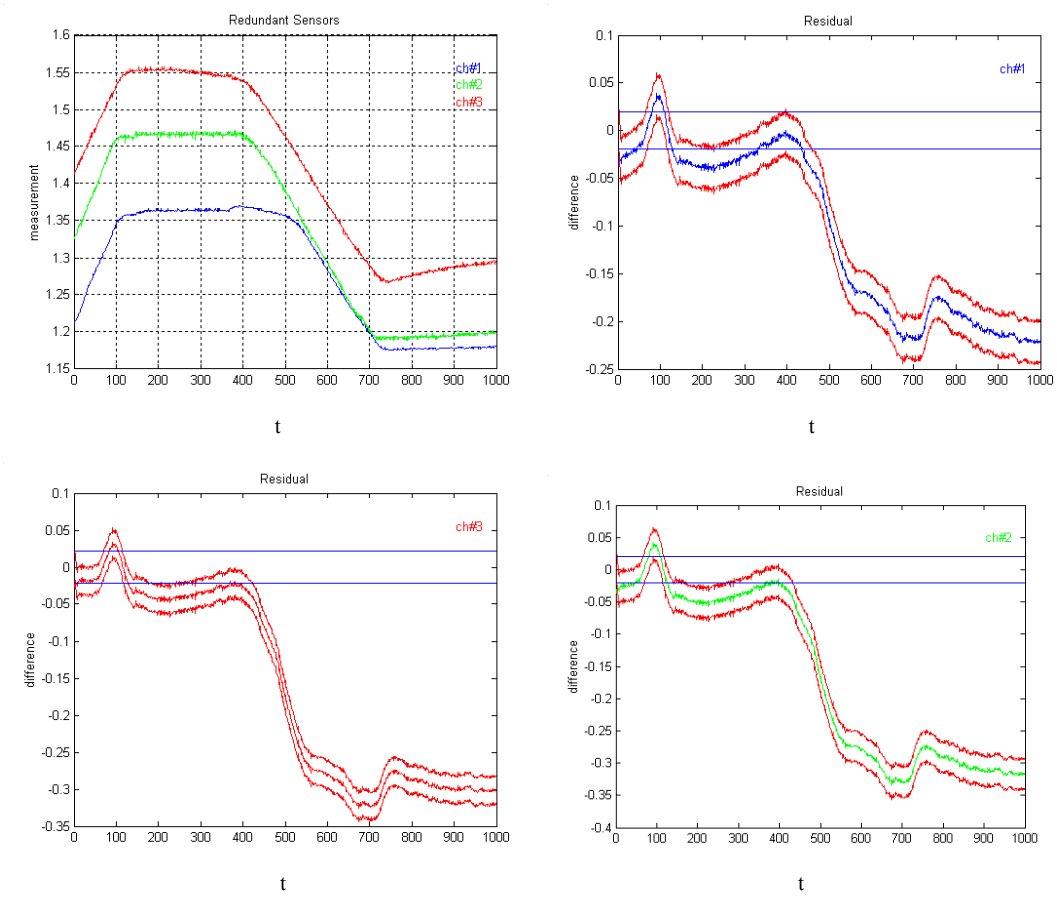

Figure 4.21. Non-stationary measurement and ICA residual.

We see that the ICA method is not applicable. Moreover, we also note that the sum of the ICA coefficient does not pass the unity check: an indication of a failure of the ICA method.

We can divide the above signal into several regions. The non-stationary regions are the ramp up region and ramp down region. In these two regions, the change is almost linear. We find that a rotation can transform the signal into a stationary signal. This gives basis for piecewise treatment of non-stationary signals.

Figure 4.22 presents the linearly increasing portion of the non-stationary signal. We see that if we apply the ICA algorithm only to that portion, the algorithm still fails. The transformation needed is to determine the slope (equation 4.3) and remove it from the signal (equation 4.4).

$$y = \alpha + \beta t$$
$$\hat{\beta} = \frac{\sum_{i=1}^{N} Y_i \cdot (t_i - \bar{t})}{\sum_{i=1}^{N} (t_i - \bar{t})^2} \tag{4.3}$$

The transform matrix A is:

$$A = \begin{bmatrix} \sqrt{\frac{1}{1+\hat{\beta}^2}} & -\sqrt{\frac{\hat{\beta}^2}{1+\hat{\beta}^2}} \\ \sqrt{\frac{\hat{\beta}^2}{1+\hat{\beta}^2}} & \sqrt{\frac{1}{1+\hat{\beta}^2}} \end{bmatrix} \tag{4.4}$$

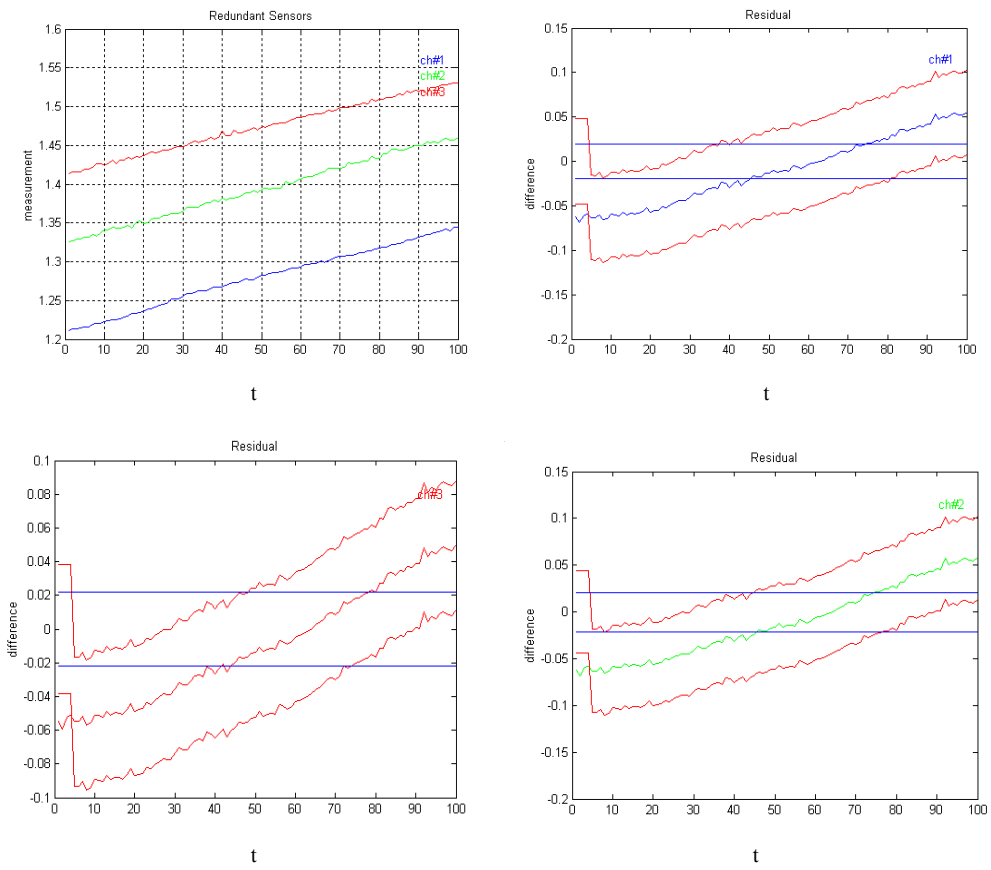

Figure 4.22. Linear region of non-stationary measurements.

For a linear transform, y = Mx

$$\int p_y(\vec{y}) \log(p_y(\vec{y}))dy = \int p_x(\vec{x}) \log \frac{p_x(\vec{x})}{|\det(M)|} dx$$

$$\Rightarrow H(\vec{y}) = H(\vec{x}) + \log(|\det(M)|) \qquad (4.5)$$

For a rotation transform, the rotation matrix satisfies the conditions of orthonormality

$$M^T M = I \qquad (4.6)$$

$$\det(M) = 1 \qquad (4.7)$$

$$H(\vec{y}) = H(\vec{x}) + \log(1) = H(\vec{x}) \qquad (4.8)$$

We see differential entropy is invariant under an orthogonal change of co-ordinates. The negentropy will also be unchanged under a rotation transform. This is the reason we can apply a rotation transform and still can find independent components.

Applying the transformation results in the signals plotted in Figure 4.23. After the transformation, we also notice that a slow drift was injected into channel 3 by heating the RTD. Then we perform ICA on the transformed space. Once the ICA estimate is found, an inverse transform is performed and the signals are plotted in their original space. This is shown in Figure 4.24.

From Figure 4.24 we see the ICA estimate contains less noise and no drift component. Thus, we are able to detect the drift using the residual plot in Figure 4.25.

This example shows that we can transform a linear non-stationary problem to a stationary problem. And ICA is applicable on the stationary signals. However, the on-line, real-time application of the transformation algorithm increases the complexity of the monitoring system significantly and is probably not practical.

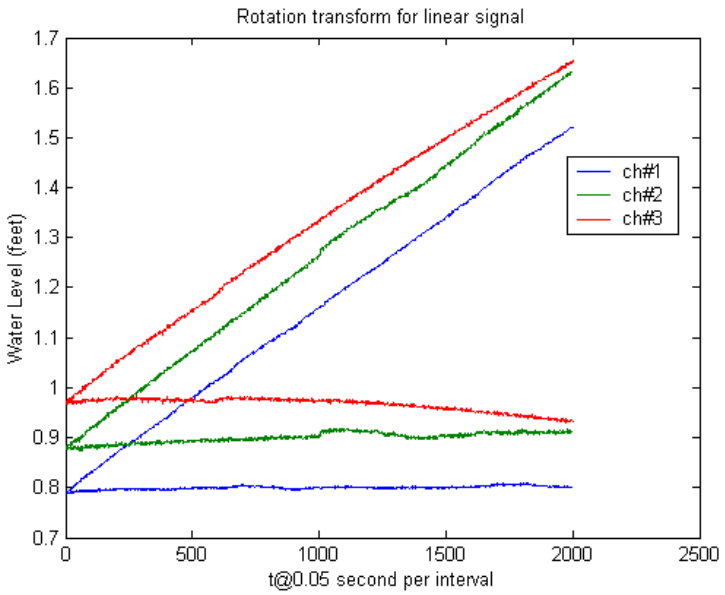

Figure 4.23. Rotated linear signal.

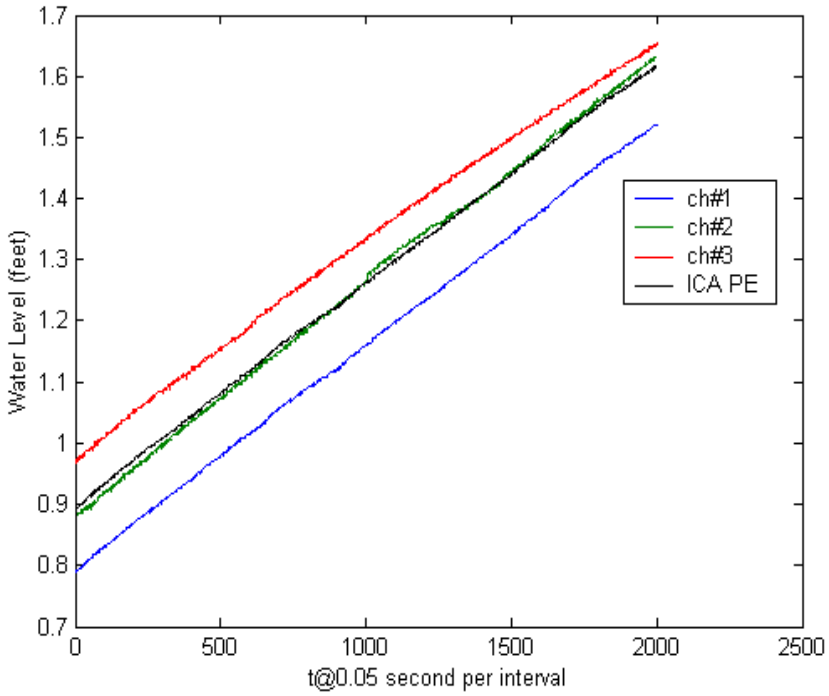

Figure 4.24. ICA parameter estimate and original measurements.

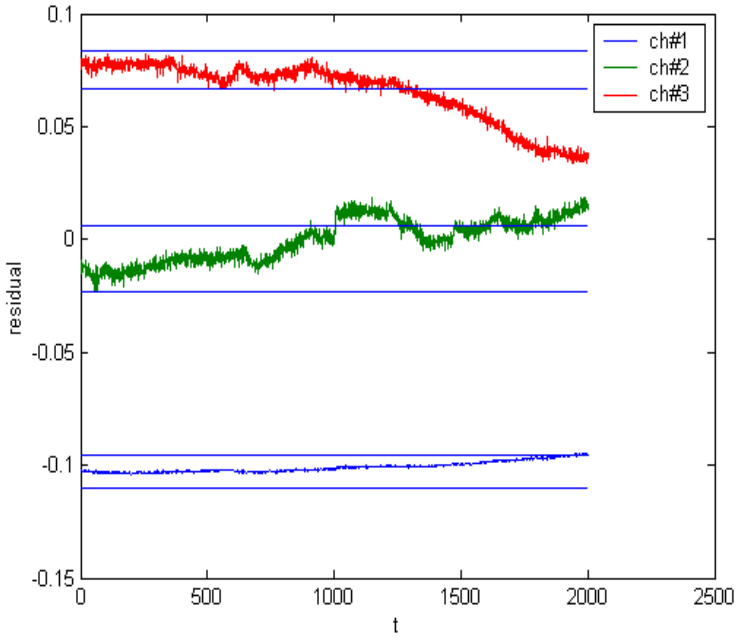

Figure 4.25. Residual of each channel.

4.8.4 Drift in a Controlled Variable

There are several cases in which only a redundancy of two is used in Nuclear Power plants. This case usually occurs when a process variable is very important but may not be a safety critical Technical Specification variable. In these cases, the variable is usually used for control and the user must choose which of the two sensors to use for control. An example of this is turbine first stage pressure in a pressurized water reactor.

In this case there are two situations that can occur when one of the two sensors starts to drift. When drift occurs in the sensor selected for control, the control system will change the actuator to keep the sensor reading constant. This will cause the actual process variable to change, but the signal used for control seems to be constant. The drift

–free sensor will change along with the process parameter. This is the unwanted situation. In the second case the drift can occur in the sensor not being used for control. Then the process variable will not be affected.

When a monitoring system is applied that creates its own estimate of the process parameter, a third case can exist. The system may be controlled by the best estimate, i.e., ICA estimate. If the ICA estimate correctly predicts the process parameter, the system will not be allowed to drift and the drifting sensor can be detected. This is investigated from SIMULINK modeling and through the water tank level control experiment.

4.8.4.1 SIMULINK Modeling

As a first experiment to investigate the application of ICA based prediction to a controlled variable, a PI controller was constructed using MATLAB's SIMULNK program (Figure 4.26). Two redundant sensors were used to measure a controlled process parameter undergoing a step response. One is then injected with a 1% ramp up to simulate a drift. If we select the non-drifting sensor as the control variable, the system response is as shown in Figure 4.27. The system quickly moves the system parameter to the desired level. If we select the drifting sensor as the control variable, the system response is shown in Figure 4.28.

In case 3, we use the ICA prediction as the controlled variable. This estimate dynamically calculates the weights of the drift and non-drift sensors. Therefore, the control variable is the best estimate of the true system parameter. The system response is as desired and is shown in Figure 4.29. In this example, the weight of sensor 1 is 0.95 and the weight of sensor 2, the drifting sensor, is 0.05.

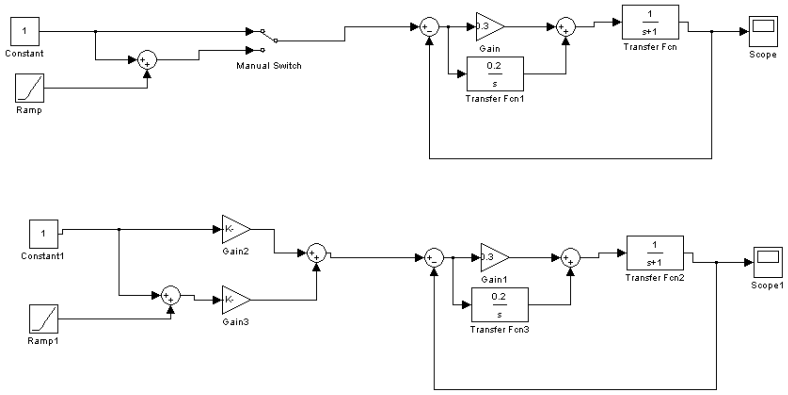

Figure 4.26. Simulink model for PI controller.

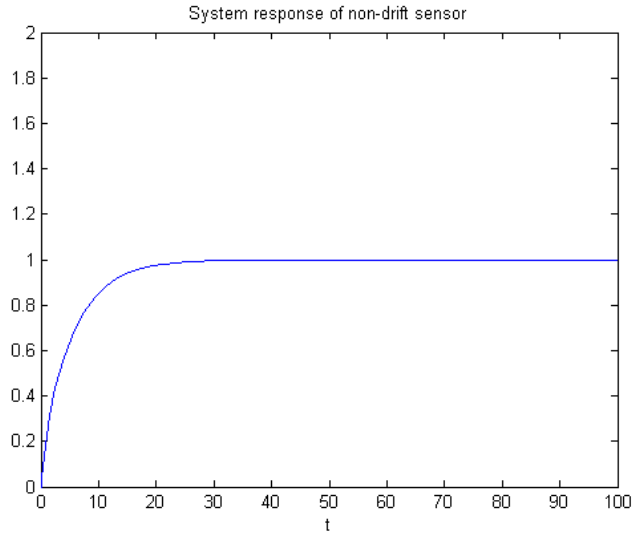

Figure 4.27. System response of non-drift control variable.

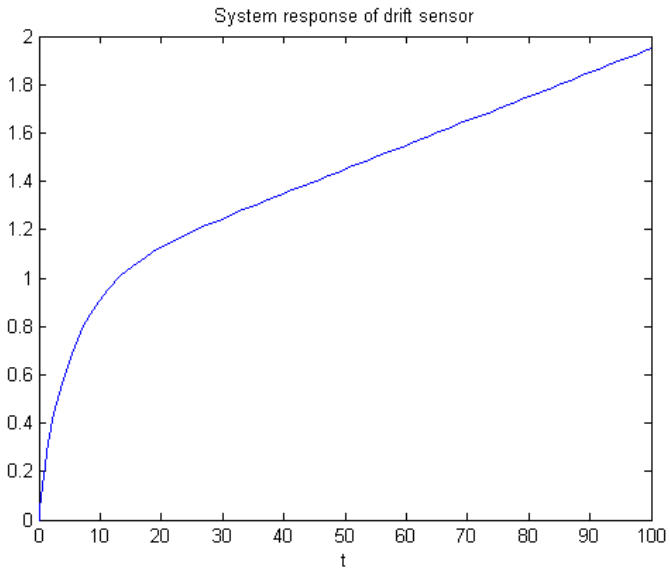

Figure 4.28. System response of drift control variable.

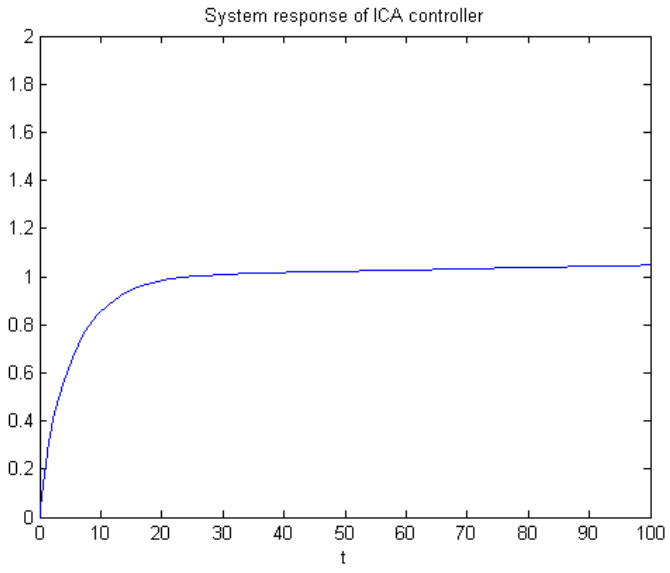

Figure 4.29. System response of the best estimate.

4.8.4.2 Experiment Using the ICA Controller

The above problem was then investigated in real time using the level control experiment. The graphical user interface of this experiment is shown in Figure 4.30. This interface allows one to choose which signal to use for control:

a). Sensor 1.

b). Sensor 2.

c). ICA Best Estimate.

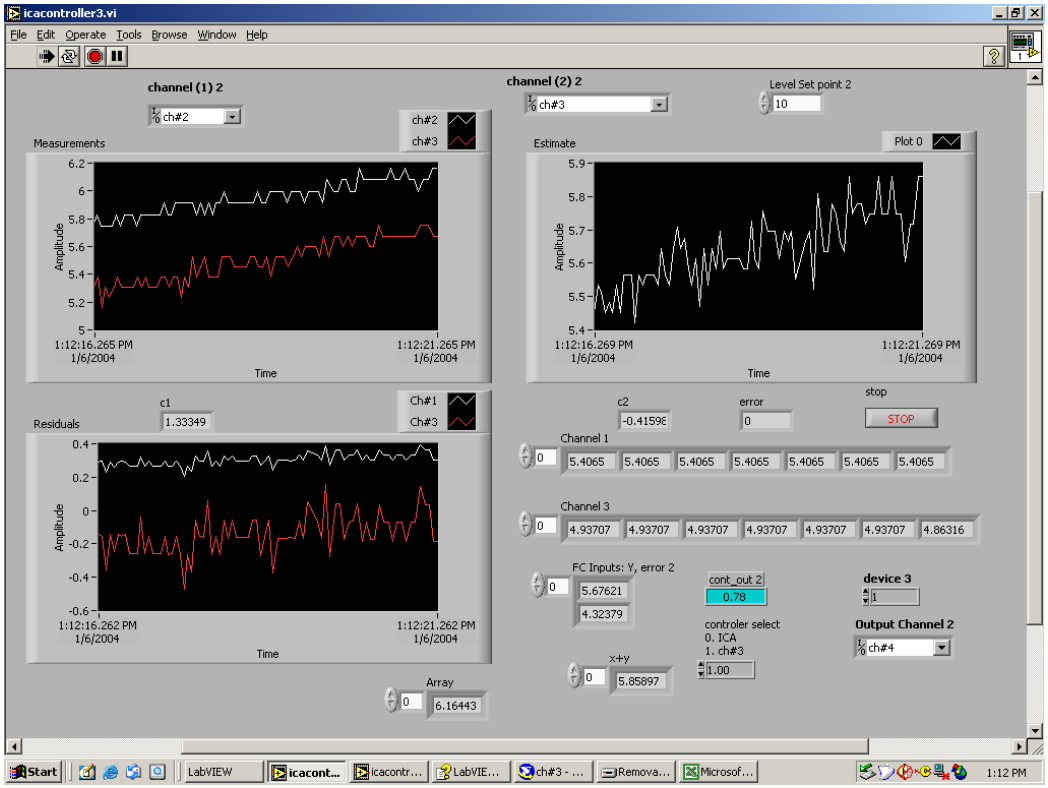

Figure 4.30. Labview ICA controller graphical interface.

In the user interface (Figure 4.30), the measured signals and their residuals are plotted on the left side, and the ICA estimate is plotted in the upper right corner. The interface allows the user to choose the controller set point. That is, it allows the user to select the desired tank level. Since we will inject the drift during steady state conditions by heating the RTD, a different set point implies a different drift rate.

The variables c1 and c2 are the ICA weights calculated by the Matlab dll for certain window sizes of the data. The system uses these two variables to calculate a unity check, which is simply the sum of the ICA weights. When the ICA algorithm is operating reliably, the two variables sum to one: the unity check passes. If the unity check does not pass, the ICA weights will automatically revert to a simple average.

Additionally, the interface allows the user to choose the control variable to be the best estimate or either of the two channels. A fuzzy controller is used to control the water level.

4.8.4.3 Experimental Results

In this example, sensor drift is injected into channel 3 at time 1240 and grows. The residuals (Figure 4.31) show that the drift is properly identified to be in Channel 3. The drift did not affect channel 1: the good channel.

In the second case, Channel 3 was selected as input to the controller with the results shown in Figure 4.32. Note that there was some bias between the two sensors before the experiment began. This bias was due to the sensors not being perfectly calibrated before the experiment. The drift is injected in channel 3 beginning at time 500.

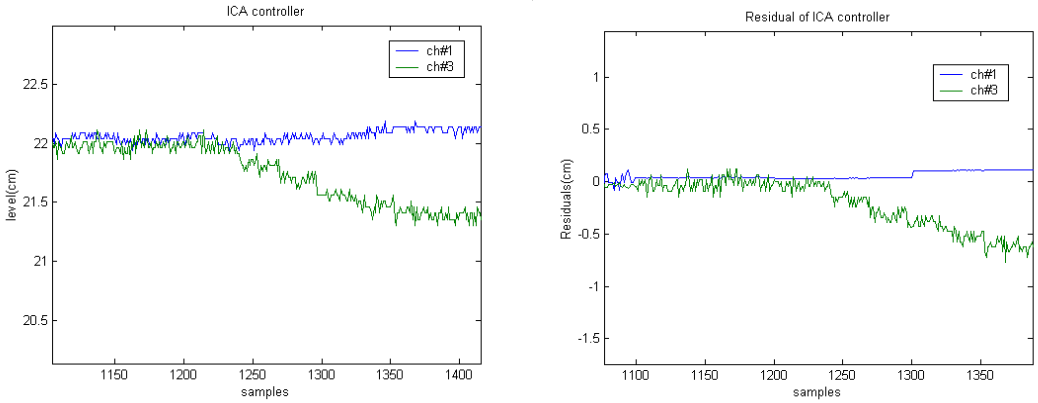

Figure 4.31. ICA controller with f=10 Hz, drift = 5% and window size = 50.

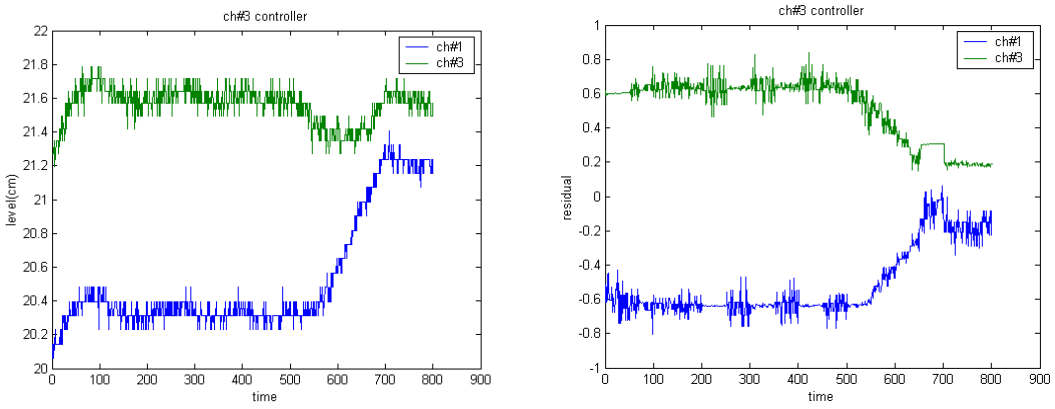

Figure 4.32. Ch#3 Controller with f=10 Hz, drift = 5% and window size = 50.

However, since the controlled sensor is drifting, the water level follows the drifting sensor. Additionally, the value of channel 1, the good channel, also increases. The measurement value of channel 3 combines the result of the drift (down) and true water level (up), and thus does not change. Finally, the residual of channel 1 goes up and the residual of channel 3 goes down. In this case, one cannot determine which channel is drifting.

4.8.5 Experiment Summary

The ICA experiment tested the ICA redundant sensor validation technique in both steady state and non-stationary conditions. During steady state, a step change fault and a slow drift fault are both detected. The fault sensor is identified even there were only two sensors. During non-stationary conditions, the ICA method is not directly applicable when the distribution in the ICA transform window becomes multi-modal. A unity check algorithm indicates the failure of ICA method. Thus, the ICA technique can still be used even when there are transients between stationary stages. The unity check works as a watchdog and it can prevent false alarms. A rotation transform algorithm was developed for linear trending signals. It could lead to a piecewise solution for general non-stationary signals.

Second, the ICA-based control strategy is demonstrated on a water level control experiment. Redundant sensors are often used to measure critical process variables that are controlled. When double redundancy is used, one must choose which of the two sensors to use for input to the controller. If a drifting or faulty sensor is chosen to be the controller input, the system will be incorrectly operated. ICA provides a solution to this

problem. The ICA prediction is robust in that the faulty component of a sensor measurement does not adversely affect the parameter estimate; even when there are only two redundant sensors. A new control strategy is presented that uses the ICA estimate as the controller input, so that drift in one of the sensors will not cause the controlled system to drift. The ICA controller is compared to a classical controller during normal and drifting conditions. The results demonstrate that in the event of sensor drifts, the system is correctly controlled and the drifting sensor is correctly identified.

4.9 Hybrid System

A hybrid system described in figure 3.1 is used by merging inferential modeling and Independent Component Analysis. The reasons to develop a hybrid system are:

1) Increase robustness,
2) Increase sensitivity,
3) Provide for common mode failure detection.

4.9.1 Data Set With Sensor Drift

The hybrid redundant sensor system is applied to data collected at a nuclear power plant that covers about one year of operation and sampled at 15 minute intervals. This data contains normal operating data and an actual pressure sensor drift. Figure 4.33 presents the data representing the two pressure sensors. A slow drift in the channel 1 measurement stating at about 13000 can be visually identified.

The PCR based inferential model is developed using 22 other variables, which are highly correlated with turbine first stage pressure. These variables and their correlations with turbine first stage pressure are listed in table 4.9.

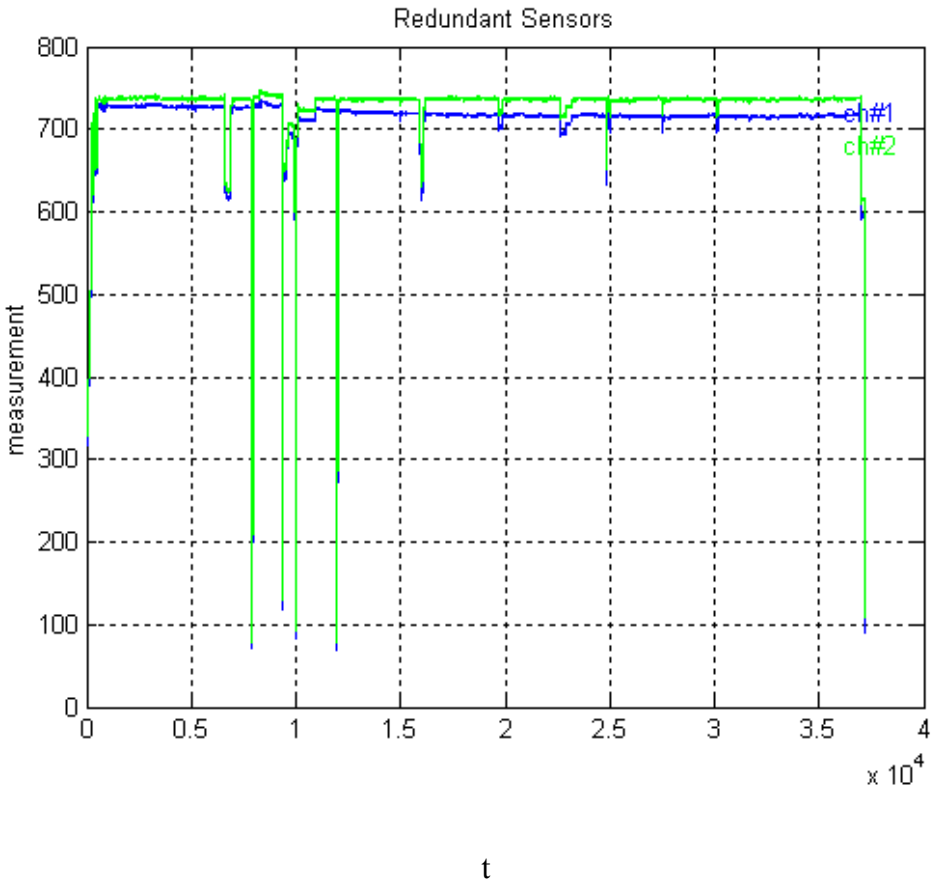

Figure 4.33. Turbine first stage pressure.

Table 4.9. Variables used in building PCR model

Variable Name	Description	Correlation with TBP1	Correlation with TBP2
x1	Gross power	0.3247	0.3434
x2	STM GEN A Feedwater flow 1	0.9882	0.9940
x3	STM GEN A Feedwater flow 2	0.9871	0.9934
x4	STM GEN B Feedwater flow 1	0.9868	0.9960
x5	STM GEN B Feedwater flow 2	0.9844	0.9953
x6	STM GEN C Feedwater flow 1	0.9941	0.9979
x7	STM GEN C Feedwater flow 2	0.9929	0.9963
x8	STM GEN A Steam flow 1	0.9869	0.9937
x9	STM GEN A Steam flow 2	0.9875	0.9940
x10	STM GEN B Steam flow 1	0.9838	0.9928
x11	STM GEN B Steam flow 2	0.9840	0.9929
x12	STM GEN C Steam flow 1	0.9938	0.9981
x13	STM GEN C Steam flow 2	0.9904	0.9946
x14	STM GEN A Steam Pressure 1	-0.8645	-0.8735
x15	STM GEN A Steam Pressure 2	-0.8401	-0.8609
x16	STM GEN A Steam Pressure 3	-0.8263	-0.8517
x17	STM GEN B Steam Pressure 1	-0.8261	-0.8493
x18	STM GEN B Steam Pressure 2	-0.8654	-0.8785
x19	STM GEN B Steam Pressure 3	-0.8237	-0.8515
x20	STM GEN C Steam Pressure 1	-0.8569	-0.8723
x21	STM GEN C Steam Pressure 2	-0.8194	-0.8436
x22	STM GEN C Steam Pressure 3	-0.6746	-0.7084

The table 4.9 shows that turbine first stage pressure is positively correlated with steam flow and feed water flow and is negatively correlated with steam pressure. This is expected because when steam flow increases, steam generator pressure will decrease and the turbine first stage pressure in the mixing chamber will go up. These correlation coefficients are strong except for variable x22. After a close look at variable x22, we find that the noise level is much higher in that channel. Turbine first stage pressure also has a fairly low correlation with the plant gross power generation. This may be partially due to system time delays. Because of this, gross power generation was not used in the inferential model.

The first 10000 data points are used as the model training set and the data from 13000 to 20000 is selected as the test set. For model training, the response variable is the average of the pressure sensor outputs. In the following graphs, the original test set subscripts are dropped new subscripts 1-7000 are used.

Figure 4.34 is a plot of the two pressure channels and the inferential sensor. It can be noted that the inferential prediction is very stable but has slightly more noise than the sensor measurements. This is primarily due to several fairly high noise sensors, such as x22, that are used as inputs. Removing these from the input space would reduce the noise but may also reduce the system robustness.

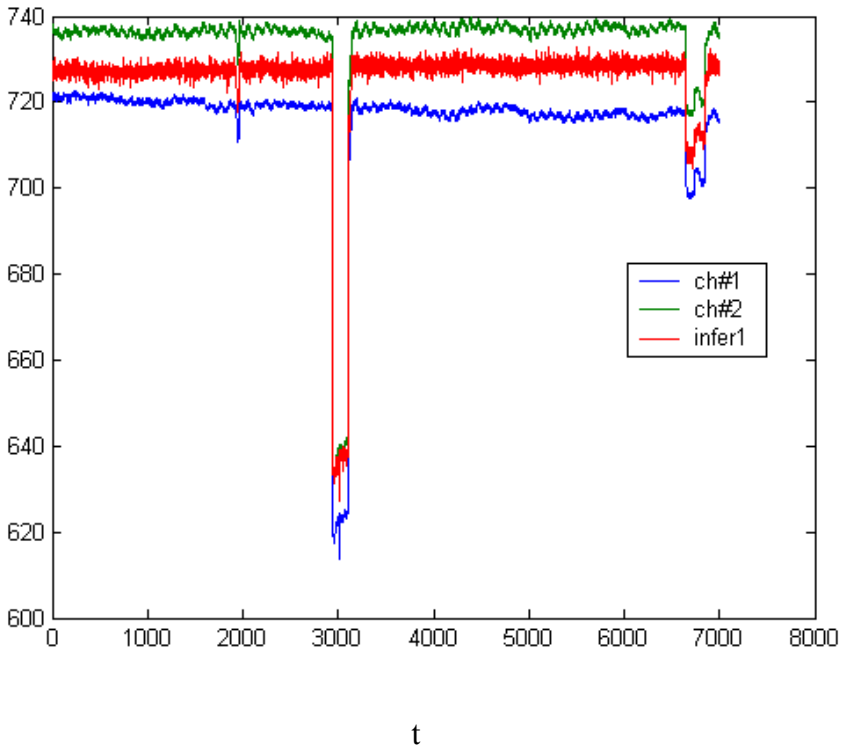

Figure 4.34. Channel 1, channel 2 and inferential sensor using PCR.

Figures 4.35 and 4.36 present the inferential senor predictions and the two sensor measurements. Figure 4.35 shows the sensor drift of Channel 1, while Figure 4.36 shows that Channel 2 is working properly. This also can be shown in the residual plots, which are presented as Figure 4.37 and 4.38. Recall the residual plots show the difference between the prediction and the sensor measurements. The residual plots are mean centered for the first 100 points so that the trend can be clearly identified.

Although the PCR based inferential sensor can identify the degrading sensor, the residual variances are high due to the noise components. These high variance residuals may make detecting very small sensor drifts difficult. This shortcoming can be remedied through application of the ICA method. A direct ICA transform of the testing data set, which includes sensor channel 1, sensor channel 2, and the inferential sensor results in a much smaller residual variance in Figure 4.39. The results are shown in Table 4.10.

Figure 4.39 presents the residuals with 95% confidence limits. It also shows the drift limits at ± 11.5 units corresponding to 2% channel drifts. Using this control chart methodology, a sensor would be determined to be out of calibration when the 95% confidence limit crosses a drift limit. If the residuals had higher noise contents, the 95% confidence intervals would be larger, and the sensor would be considered out of calibration sooner. By reducing the residual noise, greater margins are allowed resulting in more time before maintenance is required.

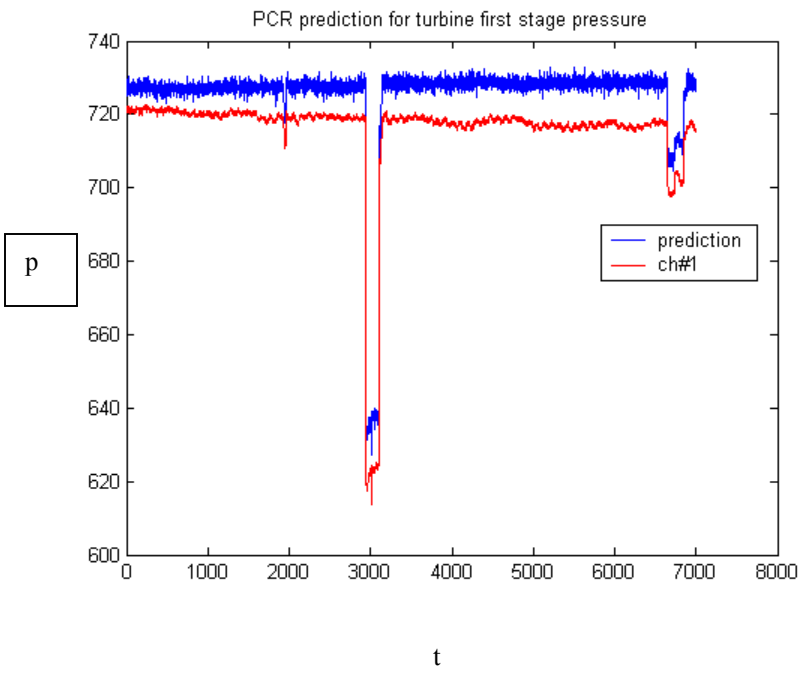

Figure 4.35. PCR and channel 1 prediction.

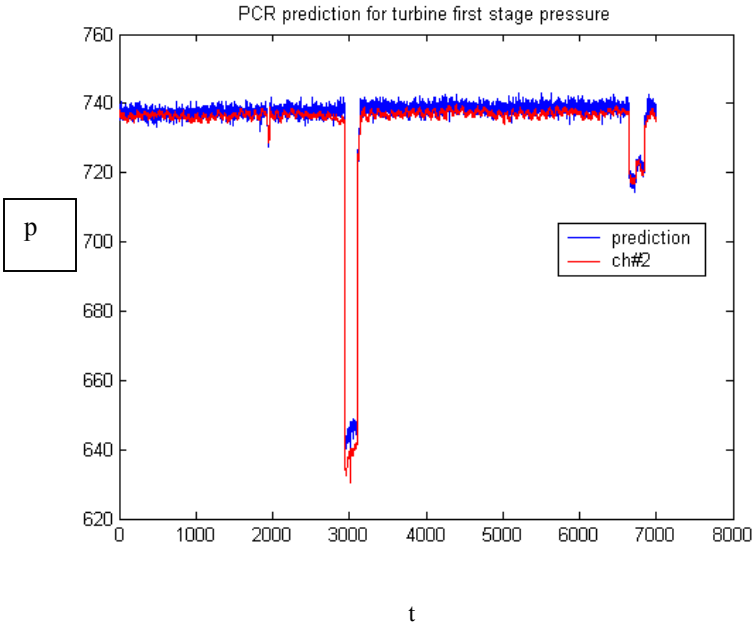

Figure 4.36. PCR and channel 2 prediction.

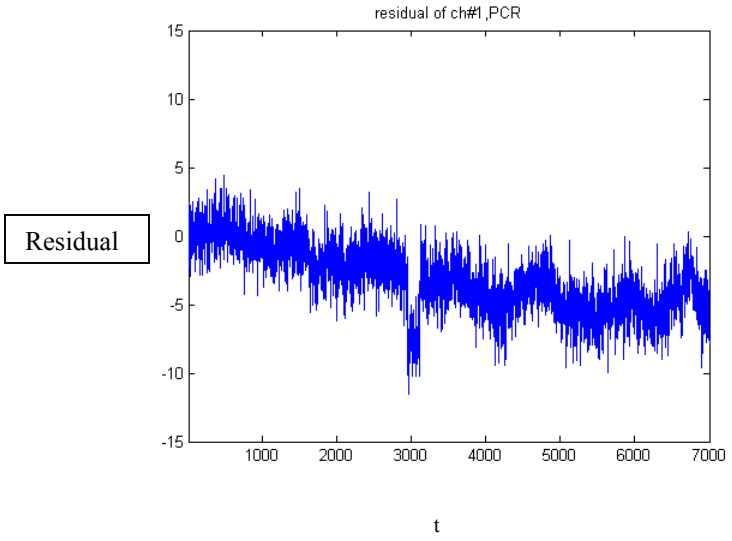

Figure 4.37. Channel 1 residual plot.

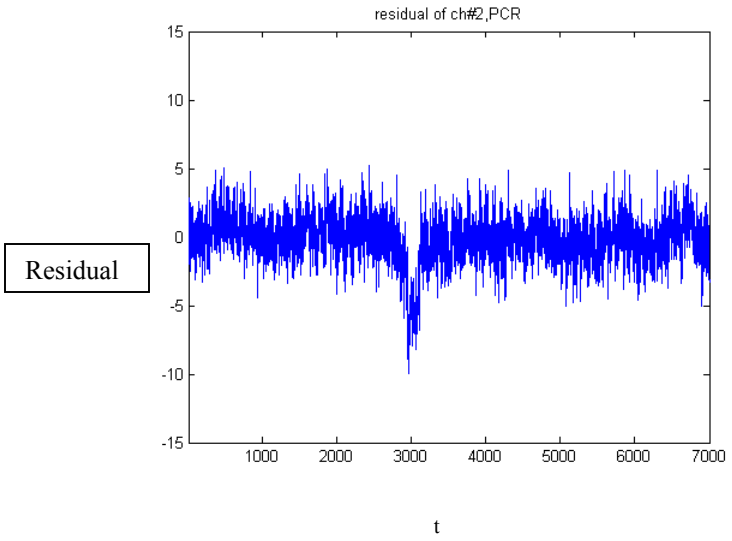

Figure 4.38. Channel 2 residual plot.

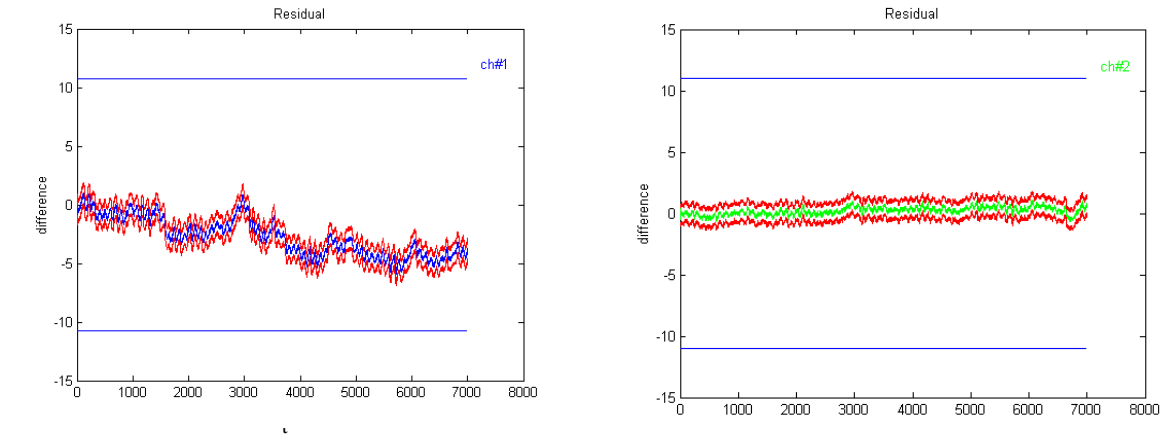

Figure 4.39. Residuals from ICA transform with inferential sensor. Red curves are 95% confidence intervals in the residuals, upper and under lines are drift limits.

Table 4.10 Variance in PCR residual and ICA + PCR residual

	PCR	ICA+PCR
Ch#1	1.1880	0.3492
Ch#2	1.3003	0.1564

4.9.2 Data Set With Anomalies

In this data set from a different Nuclear Power Plant, similar data were collected. The plant was a four loop plant and the inferential sensor used 8 feed water flow sensors, 8 steam flow sensors, and 12 steam pressure sensors for a total of 28 predictor variables. All of the correlation coefficients with turbine first stage pressure had magnitudes above 0.95. A plot of the two channels of first stage turbine pressure measurements is shown in Figure 4.40. The units of pressure have been normalized to percent power.

A PCR-based inferential model was constructed using the first 10,000 data points for training. Figure 4.41 is a plot of the inferential prediction and turbine pressure Channel 1. This figure shows several anomalous situations near sample 40,000. Upon further investigation, it was determined that four pressure sensors used as inputs to the inferential model were undergoing maintenance during the anomalous interval. Figure 4.42 shows plots of the four predictor variables that cause the poor predictions.

The type of behavior shown in Figure 4.41 is commonly termed spillover and is characterized by drifts, or other faults in predictor variables, spilling over into the response variable prediction. Most inferential models are prone to this type of behavior; however, ICA filtering can remove it. Figure 4.43 is a plot of the first stage turbine pressure prediction after using ICA to transform the two redundant channels and the PCR-based inferential prediction.

It is easily seen that the anomalous behavior is not apparent in the variable estimate and will not affect the sensor residuals. Figure 4.44 is a plot of the sensor residuals, their 95% confidence intervals and 1.4% drift limits.

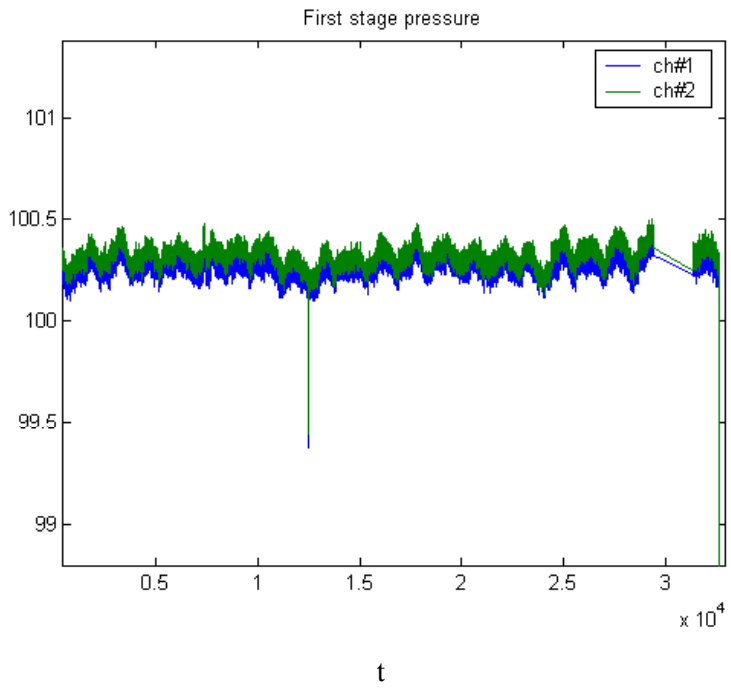

Figure 4.40. Plot of both first stage turbine pressure measurements.

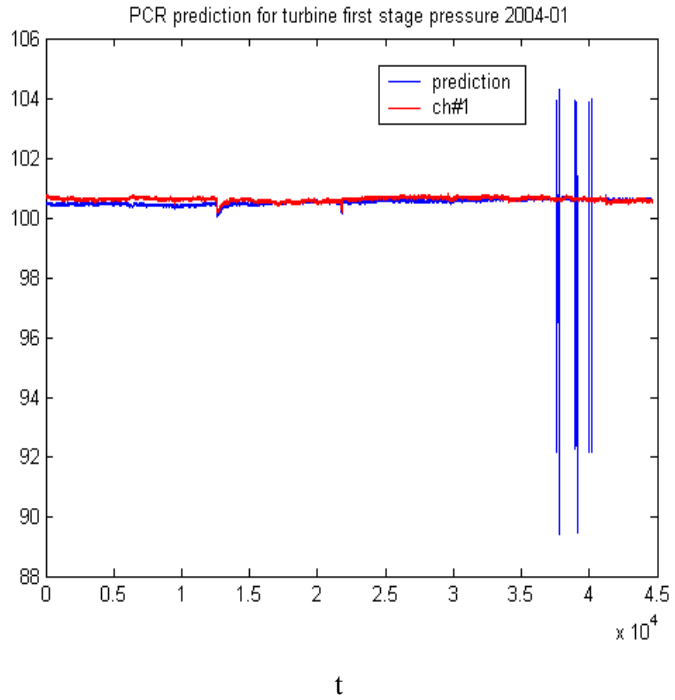

Figure 4.41. Channel 1 and the inferential prediction.

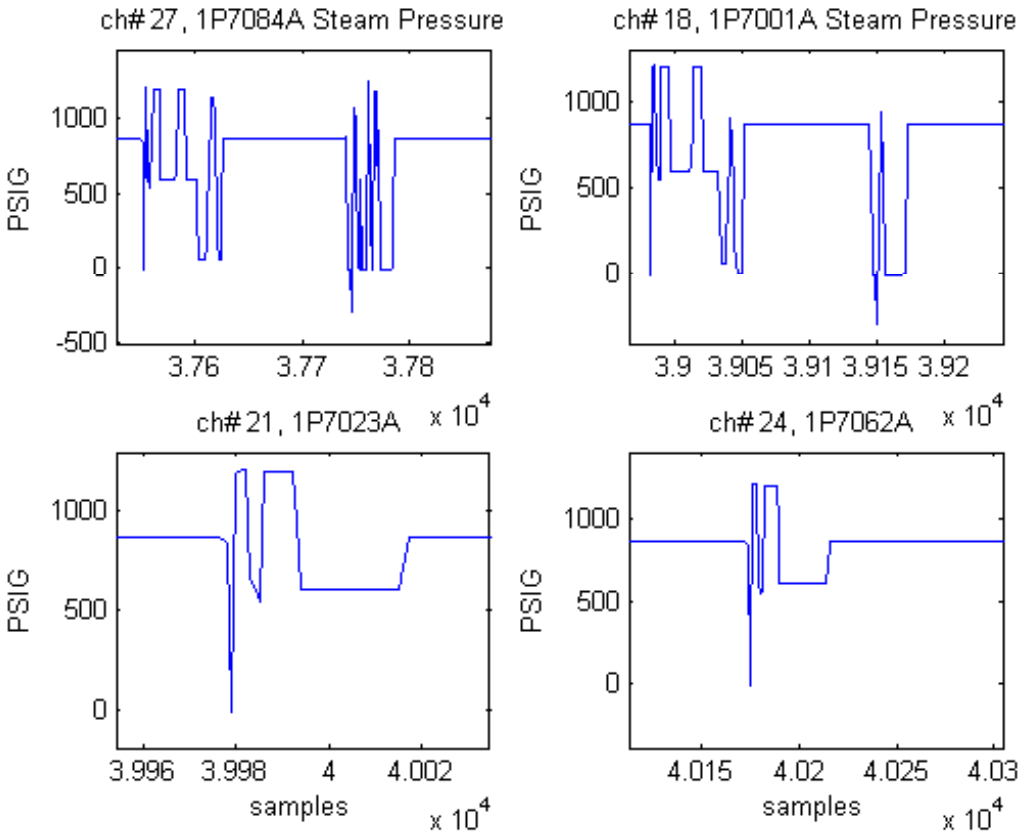

Figure 4.42. Anomalous predictor variable plots.

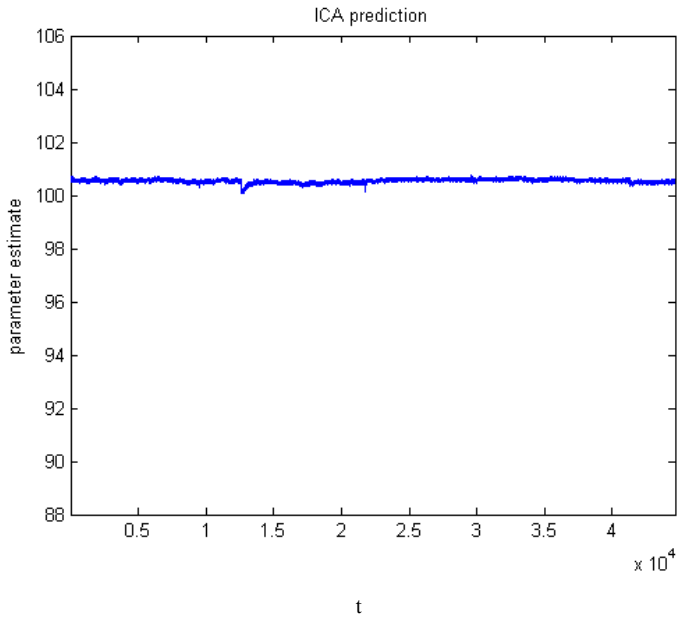

Figure 4.43. Plot of PCR with ICA prediction.

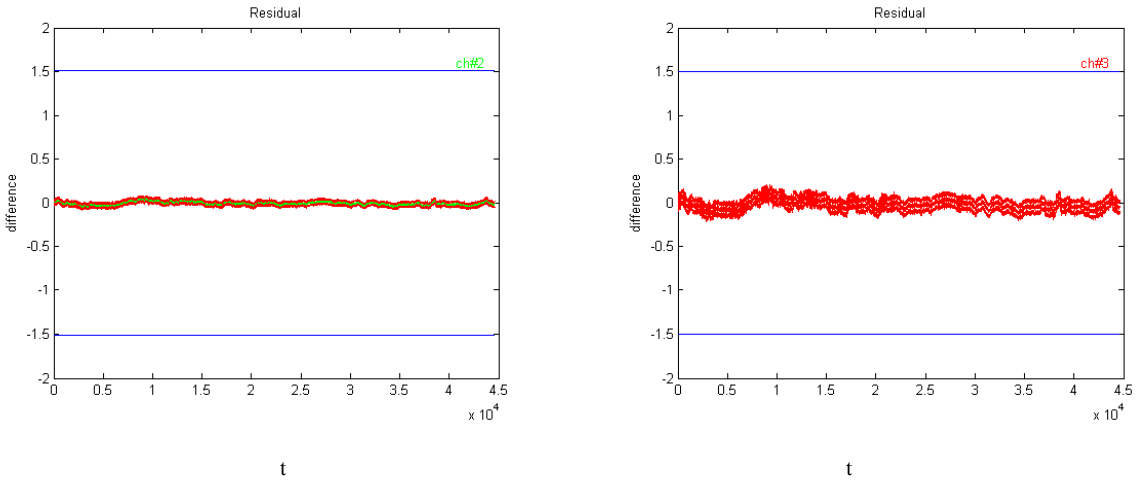

Figure 4.44. Plot of sensor residuals, 95% confidence intervals and 1.5% drift limits.

4.10 Summary

ICA has been shown to be a useful technique for modeling redundant sensors. It produces reduced noise and drift free estimates which can be used for sensor fault detection.

The ICA redundant sensor estimation technique has significant advantages over other methods commonly employed for redundant sensor calibration monitoring.

The advantages of using the ICA algorithm over the PCA and direct averaging techniques are

1. Its ability to properly model common noise and reduce false alarms.
2. Its ability to not have any spillover from faults in other channels.
3. Its ability to reduce uncertainty in the parameter estimate.

A major discovery of this research is the use of ICA to monitor systems with only two redundant channels. For the stationary signal, ICA is able to detect the faulted channel while the drift or step change is presented in one of the channels.

The ICA method is easy to use; only one parameter, the window size, needs to be specified. The moving window technique found the optimal window size captured most of the non-Gaussian component.

A reliability monitoring module was developed for the monitoring of algorithm failures. When the ICA weights fail to sum up to one, the necessary condition is invalid. Thus, one needs to switch to another averaging method. The reliability monitoring module is implemented into a real-time control system for both stationary and non-stationary signals.

A hybrid system merging ICA technique and inferential sensing technique shows its superior performance than both original systems. The hybrid system is more robust to data anomalies and also produces lower variance residuals.

CHAPTER 5
UNCERTAINTY ANALYSIS

5.0 Introduction

In this chapter the development of methods to quantify the uncertainty inherent in the ICA method is presented. A typical residual plot in Figure 5.1 can be used to explain the necessity of having accurate techniques to quantify the uncertainty. This plot shows the residual between the sensor output and the ICA prediction. Around the 500^{th} sample, the sensor begins to drift downward. This residual is bounded by lines representing 95% prediction intervals. These bounds are significant in that they contain the actual sensor drift with a 95% confidence. Also displayed are upper and lower sensor drift limits, which are represented by straight lines. When one of the 95% prediction intervals crosses the allowable drift boundary, there is less than 95% confidence that the sensor has not drifted past its allowable level. Therefore, the sensor must be scheduled for recalibration or taken out of service.

This plot shows the advantage of having predictions with small uncertainties. If the prediction uncertainty were greater than the allowable drift band, then the technique would be of no use. For the same drift rate, tighter uncertainty bands would allow extended operation before calibrations are necessary because there would be a larger cushion between the 95% prediction interval band and the drift limit.

There are several valid methods that can be used to estimate empirical prediction uncertainties. In the previous sections, the error bound was roughly estimated by computing the residual variance early in the monitoring stage.

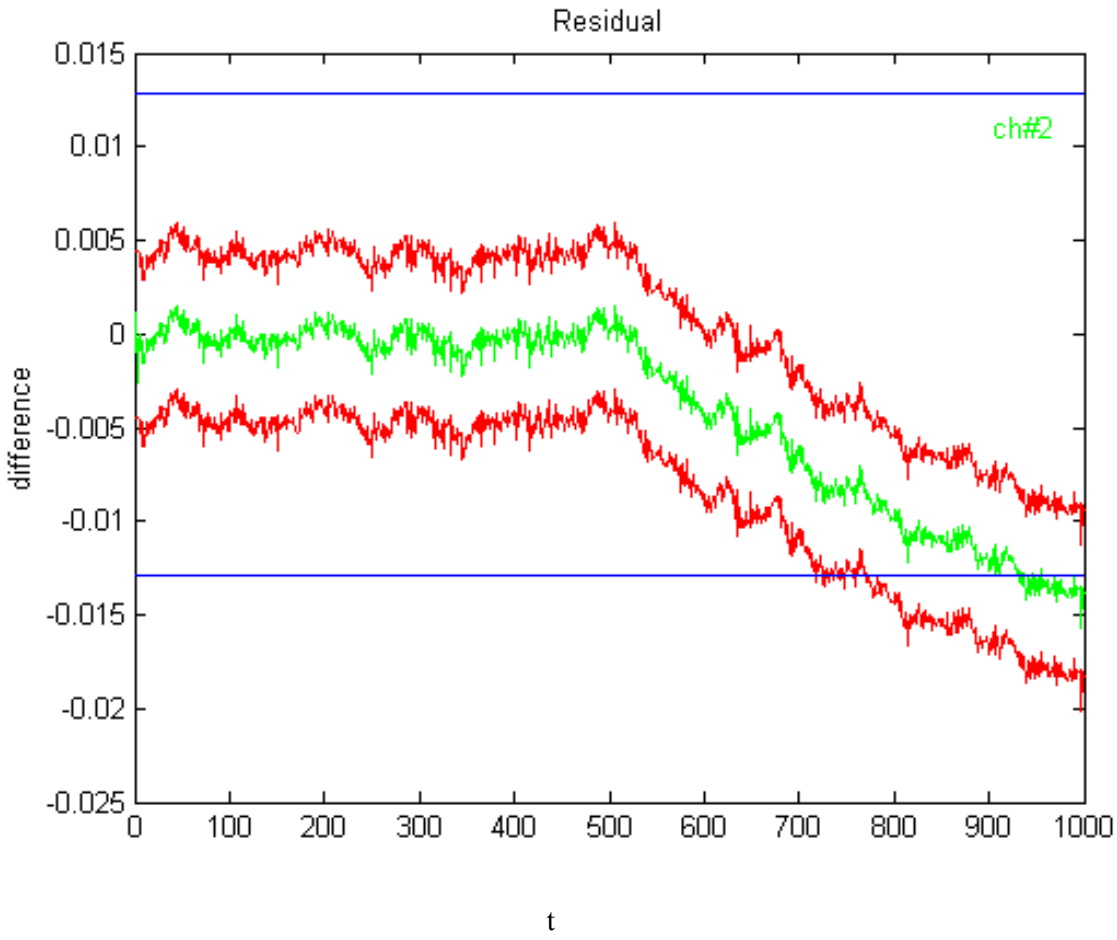

Figure 5.1. A typical residual plot with uncertainty estimate

This estimate is based on the assumption that the instrument channels are initially operating correctly and that the early residuals contain no drift component. It also assumes that the model is correctly specified so that the prediction is accurate. Thus, the residual variance can be estimated using the sample variance. In the following sections, a more precise method, called “bootstrap”, will be developed and investigated in detail.

5.1 Prediction Interval Definitions

Two terms are commonly used in relating the accuracy of a predicted parameter. These are confidence interval and prediction interval. Suppose a function of x exists, f(x), and we have some measured values of that function so that if we developed an empirical model with inputs x, we would have a desired response or targets t(x). The relationship is therefore $t(\mathbf{x}) = f(\mathbf{x}) + \varepsilon(\mathbf{x})$.

The confidence interval is defined as the accuracy of the estimate, $\hat{f}(\mathbf{x})$, to the true value, i.e. the distribution of the quantity $f(\mathbf{x}) - \hat{f}(\mathbf{x})$. The prediction interval is defined as the accuracy of the estimate to the targets (desired response values), i.e. the distribution of the quantity $t(\mathbf{x}) - \hat{f}(\mathbf{x})$. Note that even if the predictive model was perfect and its output equaled the target, t(x), it could not account for the noise $\varepsilon(\mathbf{x})$ in the targets and would not perfectly predict f(x). It can also be seen that the prediction interval encloses the confidence interval: $t(\mathbf{x}) - \hat{f}(\mathbf{x}) = [f(\mathbf{x}) - \hat{f}(\mathbf{x})] + \varepsilon(\mathbf{x})$. In this research, prediction intervals are of more practical use because they provide the accuracy with which we can predict the desired response, not just the accuracy of the model itself.

5.2 Source of Uncertainties and Bootstrap Prediction Interval

Predictive uncertainty can be decomposed into its two major components through what is commonly called the bias-variance decomposition [Hastie, 2001].

Assume $Y = f(X) + \varepsilon$, where ε is the model error, $E(\varepsilon) = 0, Var(\varepsilon) = \sigma_\varepsilon^2$. For any model prediction $\hat{f}(X)$, model error at $X = x_0$ is:

$$
\begin{aligned}
Err(x_0) &= E(Y - \hat{f}(x_0)) \\
&= \sigma_e^2 + [E(\hat{f}(x_0) - f(x_0)]^2 + E[\hat{f}(x_0) - E(\hat{f}(x_0)]^2 \\
&= irreducible\ \ error + Bias^2 + Variance
\end{aligned}
\tag{5.1}
$$

The bias term can be further decomposed into model bias and estimation bias. From equation (5.1), uncertainties have two sources: one part is from the data while the other part is from the model.

A theoretical analysis and quantification of uncertainty is difficult and usually only estimates can be made. Normally one does not know the true distribution and only asymptotic results can be given. When the model is not correctly specified, uncertainties from model misspecification are difficult to bound. Fortunately, the bootstrap method provides a technique for the empirical study of prediction uncertainty that incorporates all of these sources of error.

The bootstrap is a recently developed technique for making statistical inference using resampling methods. It requires significant computing power, but this is not a concern for calculations performed off-line on current computers. The bootstrap technique for estimating standard error was developed by Efron in 1979. The basic technique involves sampling an empirical distribution with replacement. The bootstrap algorithm begins by sampling the original data set with replacement resulting in a bootstrap sample $\mathbf{x^*} = (x_1^*, x_2^*, \ldots, x_n^*)$. The sample contains n randomly sampled observations. This is repeated a large number of times resulting in B independent bootstrap samples: $x^{*1}, x^{*2}, \ldots x^{*B}$, each of size n. Typical values for B, the number of bootstrap samples, range from 50 to 200 for fairly accurate standard error estimates [Efron, 1993].

For each of B bootstrap samples, a statistic is calculated: $s(x^{*b})$. For example, if s(x) is the sample median, then $s(x^{*})$ is the median of the bootstrap sample. The bootstrap estimate of the standard error is the standard deviation of the statistic calculated from each bootstrap sample.

$$s\hat{e}_{boot} = \sqrt{\frac{\sum_{b=1}^{B}[s(x^{*b}) - \frac{\sum_{b=1}^{B} s(x^{*b})}{B}]^2}{B-1}} \qquad (5.2)$$

For large n, one can obtain a bootstrap confidence interval assuming a normal distribution [Efron, 1981, 1987].

Assuming any estimate $\hat{y} \sim N(y, se^2)$,

$$Z = \frac{\hat{y} - y}{se} \sim N(0,1)$$

$$\Pr ob\{\theta \in [\hat{y} - z^{(1-\alpha)} \cdot se, \hat{y} - z^{(\alpha)} \cdot se]\} = 1 - 2\alpha \qquad (5.3)$$

The same analysis can be applied to construct a prediction interval.

The bootstrap method is straight-forward to implement on a computer. It commonly has a computational advantage over asymptotic theory due to its simplicity. Additionally, the convergence rate is better than asymptotic theory under certain distributions. In some cases, there are no alternatives and bootstrap is the only method capable of calculating uncertainty statistics [Schimek, 2000].

5.3 Bootstrap Prediction Interval Example

We will present a sample bootstrap application. Suppose we have a predictor variable, X ~ N (0,1) and response variable Y, which is a linear function of X plus some Gaussian noise:

$$Y = 2 * X + 0.5 * norm(0,1) \qquad (5.4)$$

From equation (2.18), a least square regression coefficient is estimated. The predicted value is calculated from equation (2.19). The least square model is shown in Figure 5.2. The regression line goes through center of most of the data points.

From equation (2.24), the confidence interval of the regression coefficient is calculated:

$$\hat{\beta} \sim N(\beta, (X^T X)^{-1} \sigma^2) \sim N(2, 0.087) \qquad (5.5)$$

In reality, the true value of β is unknown but can be estimated using bootstrap method.

First, 100 models are calculated from a resampled distribution. A bootstrap confidence interval is obtained by taking the mean and standard deviation of the regression coefficient:

$$\hat{\beta} \sim N(2.0017, 0.056) \qquad (5.6)$$

The bootstrap confidence interval of 0.056 is a little smaller than the analytical result (0.087) in the above example. This can be due to the finite random numbers generated.

The same bootstrap procedure is used to calculate a prediction error and a prediction interval. Recall that the prediction error is the difference between the prediction estimate and the true value. Since the true value is unknown, the prediction error is estimated by the bias and variance of the residuals where the residuals are the difference between the model prediction and sampled value. If the model is unbiased, the

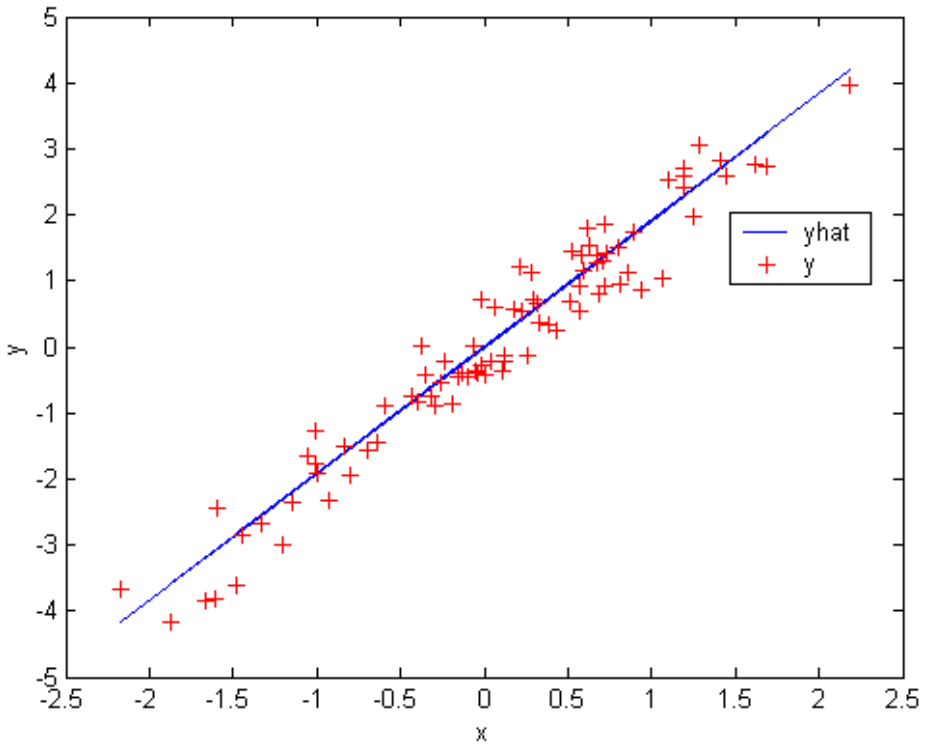

Figure 5.2. Linear model between predictor and response variable

mean of the residuals is equal to zero. In this case, the variance of the residuals is the estimate of prediction error.

The bootstrap residuals are shown in Figure 5.3. For each of the 86 data points, 100 models are calculated and the mean residual and the residual variance is plotted.

The average value of the residual means is –0.0031 (near zero), giving evidence that the linear model prediction is unbiased; hence, the prediction error can be estimated by the residual variance. The average of the residual standard deviations is 0.5. Thus, the

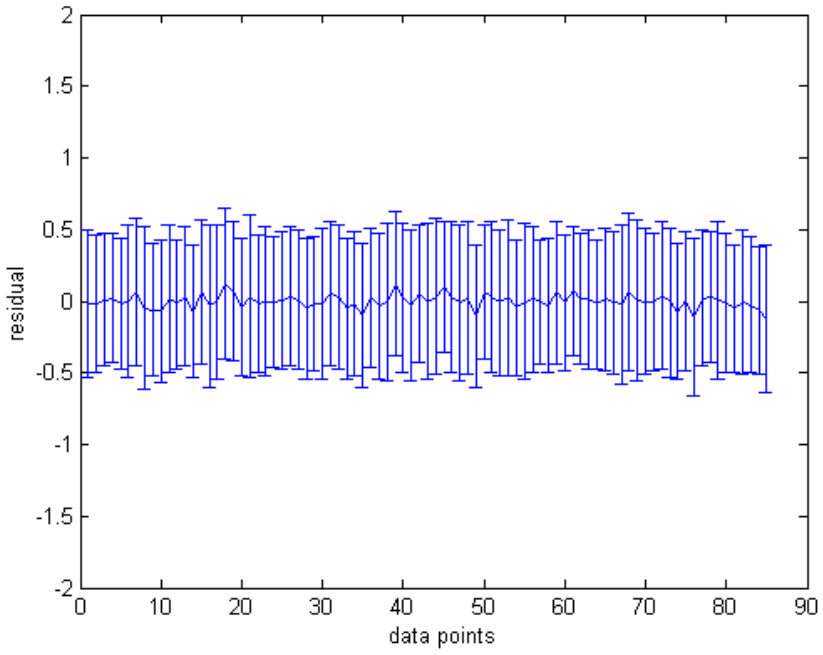

Figure 5.3. Plot of residual means and standard deviations for each data point.

estimate of the prediction error is 0.5, which is exactly the noise level in equation 5.4. The 95% prediction interval will be:

$$[\hat{y} - 1.96 * \hat{\sigma}, \hat{y} + 1.96 * \hat{\sigma}] \qquad (5.7)$$

The 95% prediction intervals calculated by the bootstrap method are plotted in figure 5.4.

As a second example, the noise level of equation (5.4) was increased to 0.9.

$$Y = 2 * X + 0.9 * norm(0,1) \qquad (5.8)$$

The Mean and average standard deviation of the residuals are -0.0039 and 0.89 respectively. Again, the estimate of the prediction error is equal to the noise level in equation (5.8).

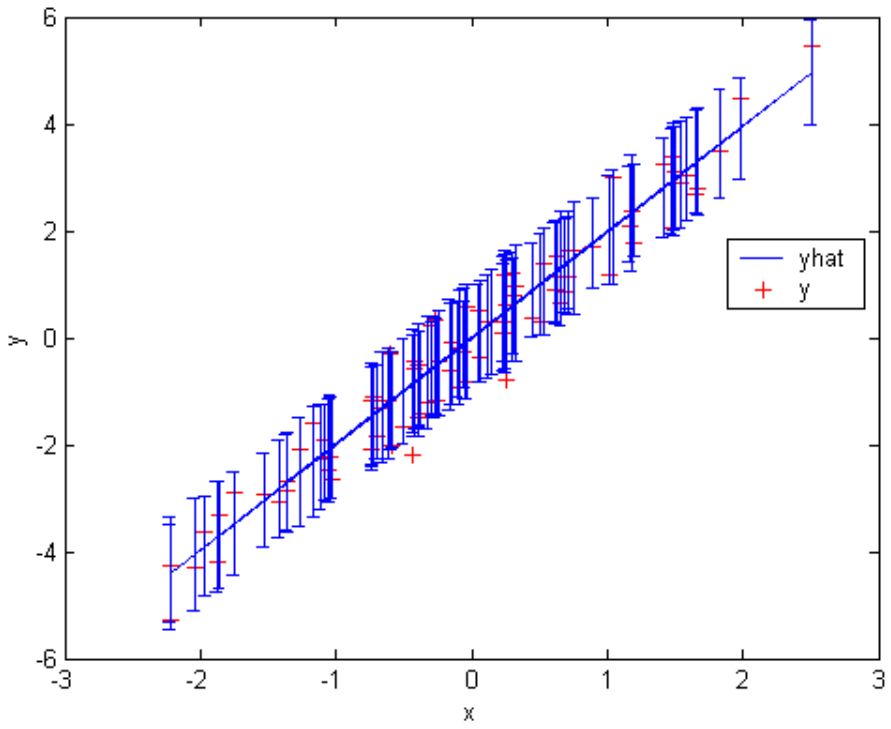

Figure 5.4. Linear model predictions and 95% prediction intervals using bootstrap.

For the above examples, the linear model prediction is unbiased. The contribution of prediction error from the bias term is close to zero. The following example shows the prediction interval from a biased prediction.

For the function Y and X ~ N (0,1),

$$Y = X * X + 0.5 * norm(0,1) \qquad (5.9)$$

A linear model is calculated to estimate the nonlinear function in equation (5.9). A plot is shown in Figure 5.5. Bootstrap prediction bias and variance are shown in Figure 5.6 and the prediction interval is shown in Figure 5.7.

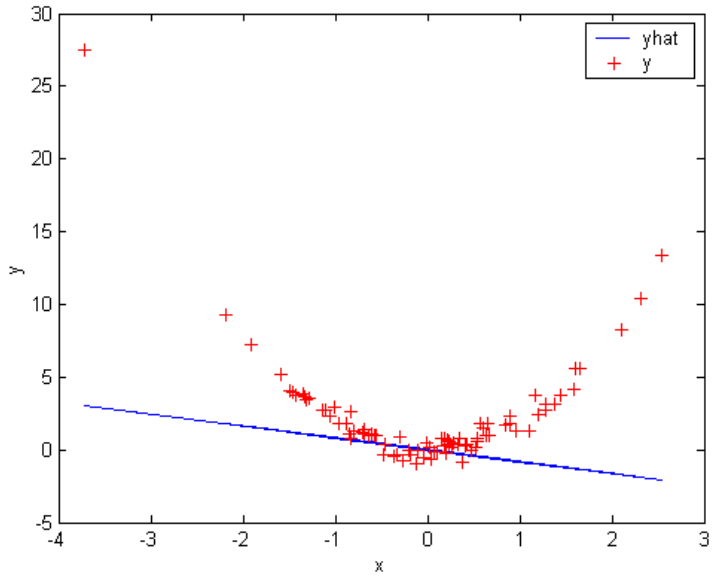

Figure 5.5. Linear estimate for a nonlinear function.

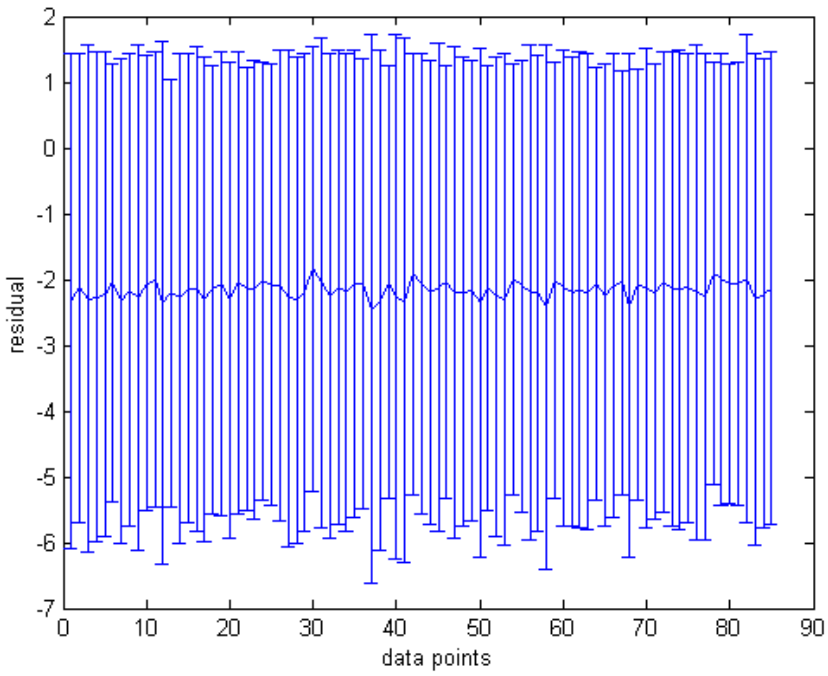

Figure 5.6. Bootstrap bias and variance for the biased model.

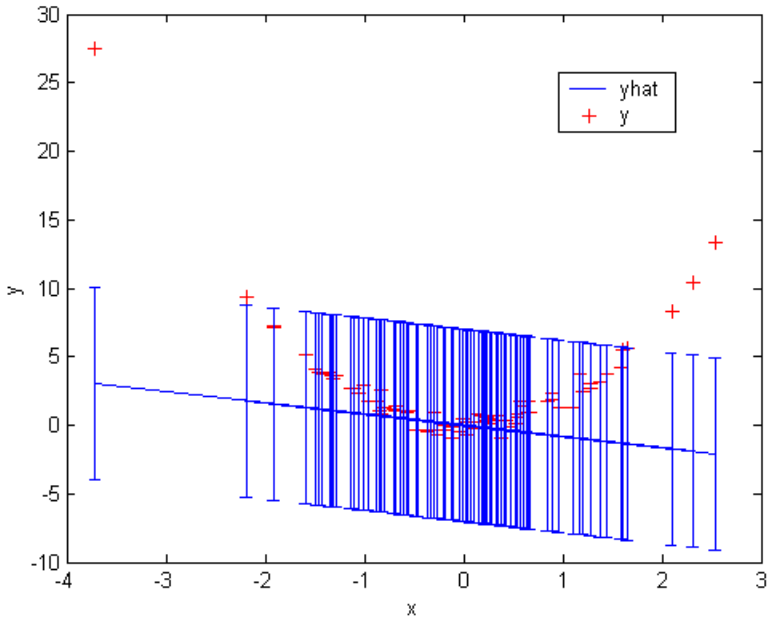

Figure 5.7. 95% prediction interval for a biased model

This example shows that a linear model is not a good estimate for a nonlinear function. However, the bias term can be estimated from bootstrap method. Thus, the 95% prediction interval is large and covers approximately 95% of the samples. The bootstrap method to estimate the prediction interval now is extended to ICA uncertainty analysis.

5.4 ICA Prediction Error

The ICA model is derived from the non-Gaussianity principle. Because ICA is a data-driven transformation method, the ICA prediction error largely depends on the data set and the model assumptions. The bootstrap method provides a technique to directly compute an estimate of the prediction errors.

Similar to the example in Section 5.3, the prediction error is estimated from the bootstrap residuals. For a given data set (i.e., redundant sensor measurements), an ICA transform is applied to the bootstrap sample sets. For each bootstrap sample set, an ICA parameter estimate is produced and residuals are generated from that calculation. This procedure is repeated B times, with B being as large as 100. Thus for each data point, 100 residuals are generated. The mean of the residuals is calculated from equation (5.10) and standard deviation of the residuals from equation (5.2).

$$\hat{y} = s(x^{*b}) = \frac{\sum_{i=1}^{B} x_i}{B} \tag{5.10}$$

The total prediction error is estimated by combining the bias and the variance estimates:

$$\hat{\sigma} = \sqrt{(\hat{y})^2 + (s\hat{e}_{boot})^2} \tag{5.11}$$

The 95% prediction error interval is

$$\left[\hat{y} - 1.96 \cdot \hat{\sigma}, \hat{y} + 1.96 \cdot \hat{\sigma}\right] \tag{5.12}$$

The bootstrap method estimates the bias term and variance terms of the prediction error simultaneously.

5.5 Bootstrap Prediction Example for ICA

The bootstrap method for ICA prediction error is presented through the following example, which uses ICA to compute the sensor estimate using two redundant sensor inputs and an inferential model prediction. Figure 5.8 presents the same data from Section 4.9.2, which is a measurement of nuclear power plant first stage turbine pressure.

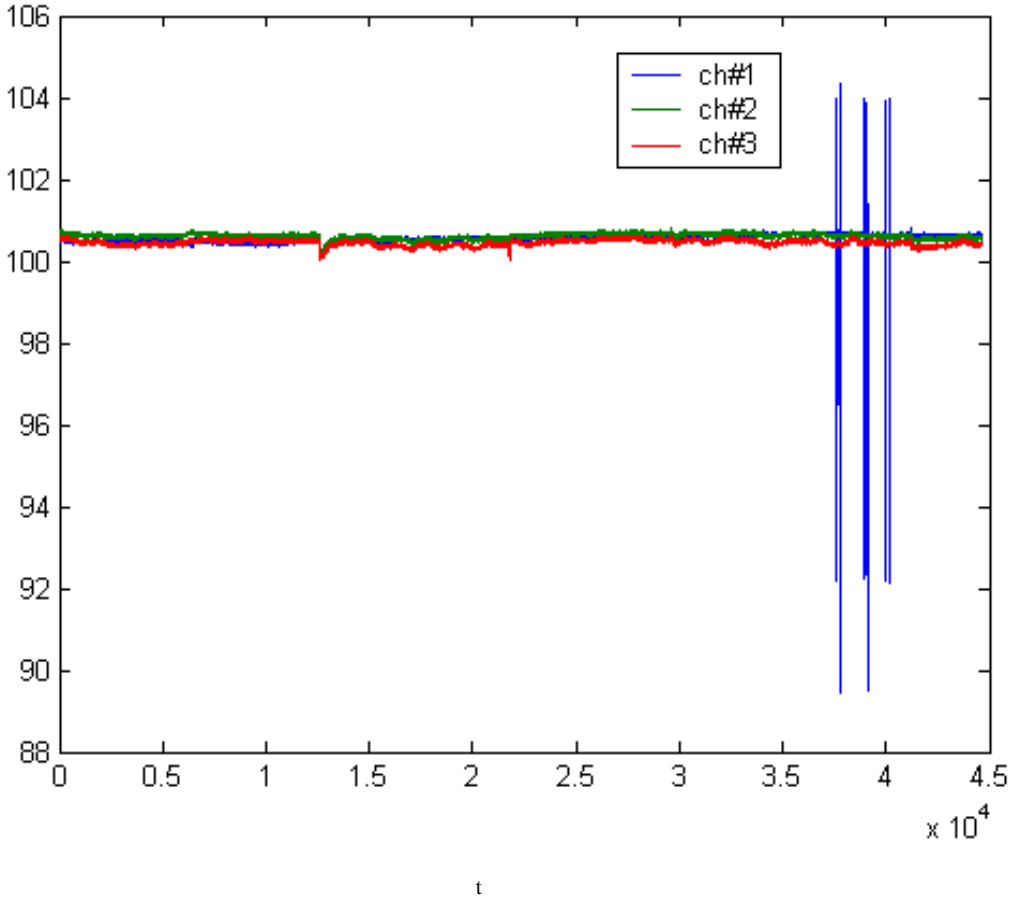

Figure 5.8. First stage pressure measurements with inferential sensor (ch#1).

Channel 1 is the PCR inferential prediction calculated from a model using 28 predictor variables from previous measurements. Channels 2 and 3 are the two redundant sensor measurements.

It is obvious that the inferential sensor prediction contains data anomalies. Figure 4.43, shows that the ICA model properly filtered these anomalies and produced a fairly constant, noise-free, parameter prediction. However, these anomalies do produce a higher bias term as will be shown.

The ICA weights were calculated with 1000 bootstrap samples and are shown in Figure 5.9. The PCR prediction weights are mostly near zero except for several small spikes. This means that the PCR model prediction is not weighted heavily, and therefore not used to a high degree in the model. This is because the PCR prediction has a higher variance due to the data anomalies. Because the bootstrap procedure uses observation vectors that are slightly differently for each sample, when the data anomalies are not included in bootstrap samples, the ICA calculates the weights differently and gives the prediction more weight.

The ICA parameter prediction is a linear combination of channel 1 (PCR inferential prediction), and redundant sensor channels 2 and 3. A slightly different weight for each channel produces a slightly different parameter prediction for each bootstrap

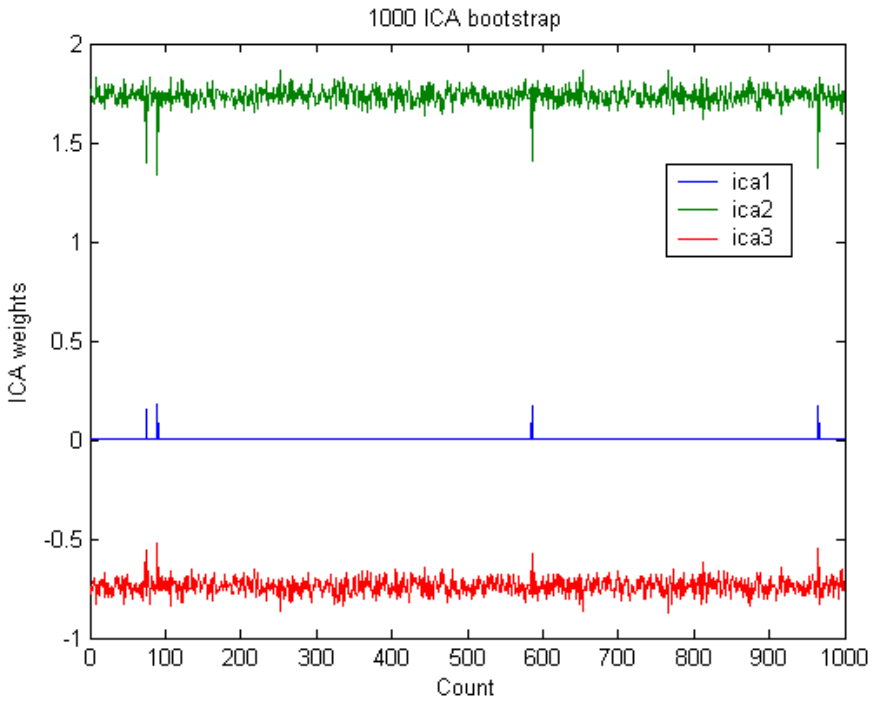

Figure 5.9. ICA weights for 1000 bootstrap samples

sample. Thus, the residuals between the ICA prediction and the channel measurement are slightly different each time. Uncertainties are estimated from equation (5.2), (5.10) and (5.11).

A histogram plot of the residuals for one data point is shown in Figure 5.10. It is evident that the distribution is fairly normal so we can use 1.96 times the total prediction error as an estimate for the 95% prediction intervals.

The mean and variance of the residuals for the first 25 data points are plotted in Figure 5.11. Recall that the mean is an estimate of the bias. Actually, it includes the irreducible error due to the noise in response variable, but treating it as a bias is a conservative assumption. The bias term is much higher than the variance term in this case and the major contributor to prediction error is from the bias.

The total prediction error is calculated from Equation (5.11) and plotted in Figure 5.12. The expectation of prediction error is 0.078, which is about 0.08% of measurement. The estimate of residual variance used in Figure 5.1 is 0.0835. These two estimates are very close to each other, so for this case, the simple technique of estimating the prediction error discussed in Chapter 4 was valid.

The prediction interval is calculated from equation (5.12). A 95% prediction interval with the ICA estimate is shown in Figure 5.13. Channel 2 is within ICA prediction interval limits, which show no drift in the channel. The prediction intervals are applied to the residuals and are shown in Figure 5.14 along with 1.4% drift limits. The uncertainties are small enough that the sensors have significant room to drift before they would need to be scheduled for calibration.

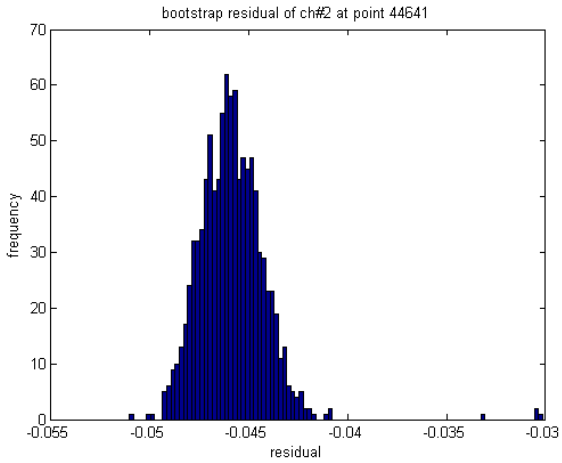

Figure 5.10. Histogram of 1000 bootstrap residuals for data point 44641.

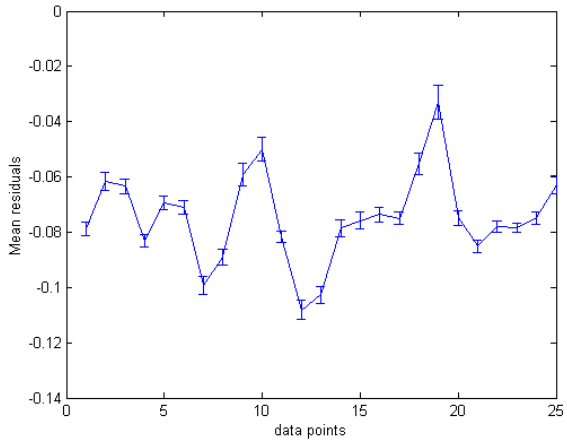

Figure 5.11. Mean and variance of residuals for 1000 bootstrap samples of first 25 data points.

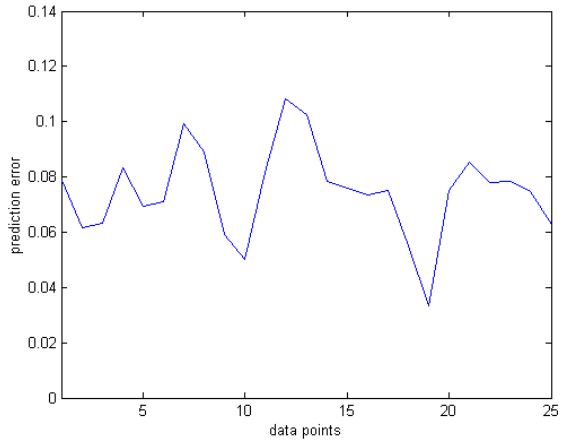

Figure 5.12. Total prediction error for 1000 bootstrap samples of first 25 data points.

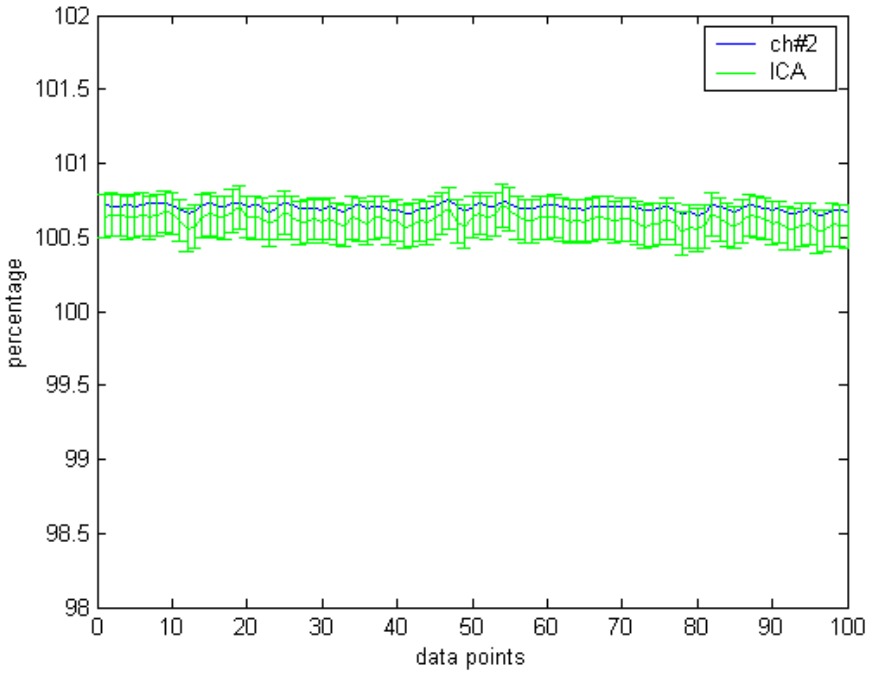

Figure 5.13. ICA prediction and 95% prediction intervals.

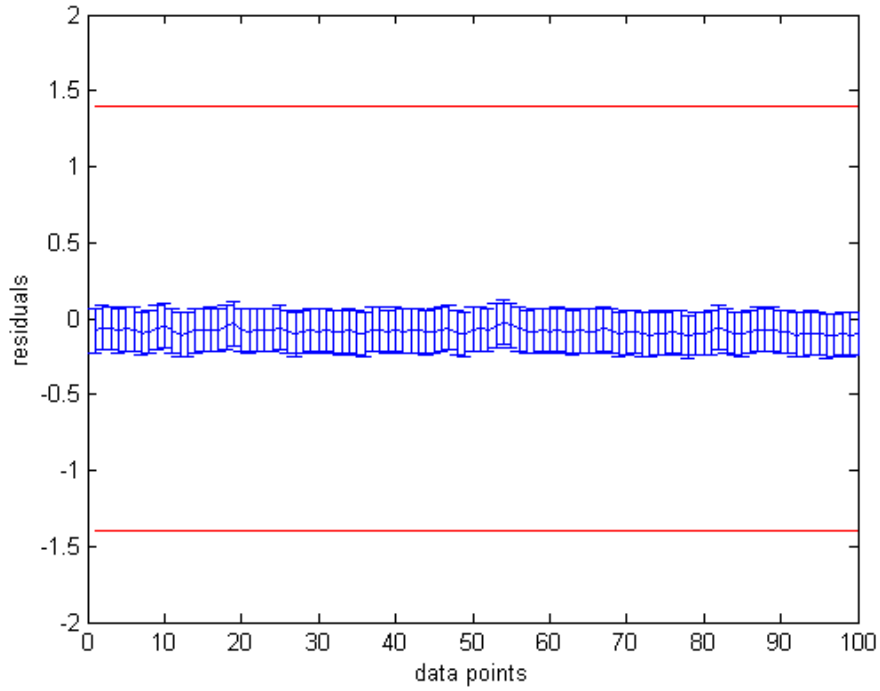

Figure 5.14. 1.4% drift limit and residuals with calculated prediction intervals.

5.6 Prediction Interval Comparison

It was shown in section 4.9.1 that hybrid ICA incorporating a PCR inferential sensor produces lower variance predictions than PCR model alone. From the comparison of prediction interval, this feature is shown clearly.

The same dataset from Section 5.5 is used for calculation. For the PCR model, a similar bootstrap procedure is used to build 200 models from training data. Bias and variance are calculated from residuals between PCR predictions and channel 2 responses on the testing data. The histogram of the prediction error is shown in Figure 5.15. The PCR plus ICA result is shown in Figure 5.16.

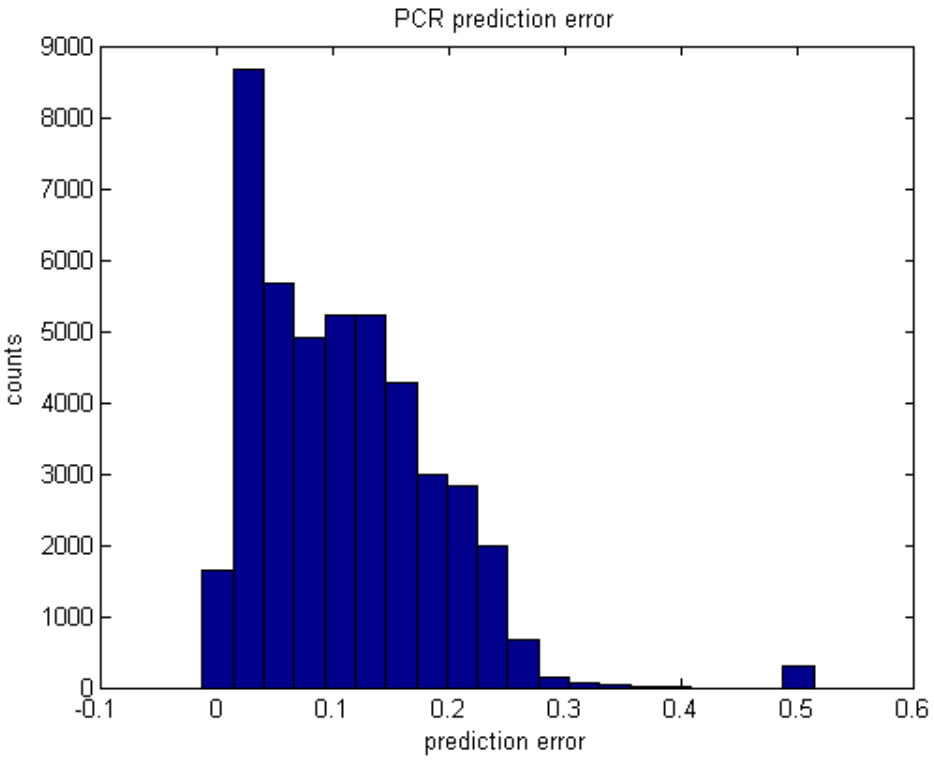

Figure 5.15 Histogram of bootstrap prediction error for PCR model.

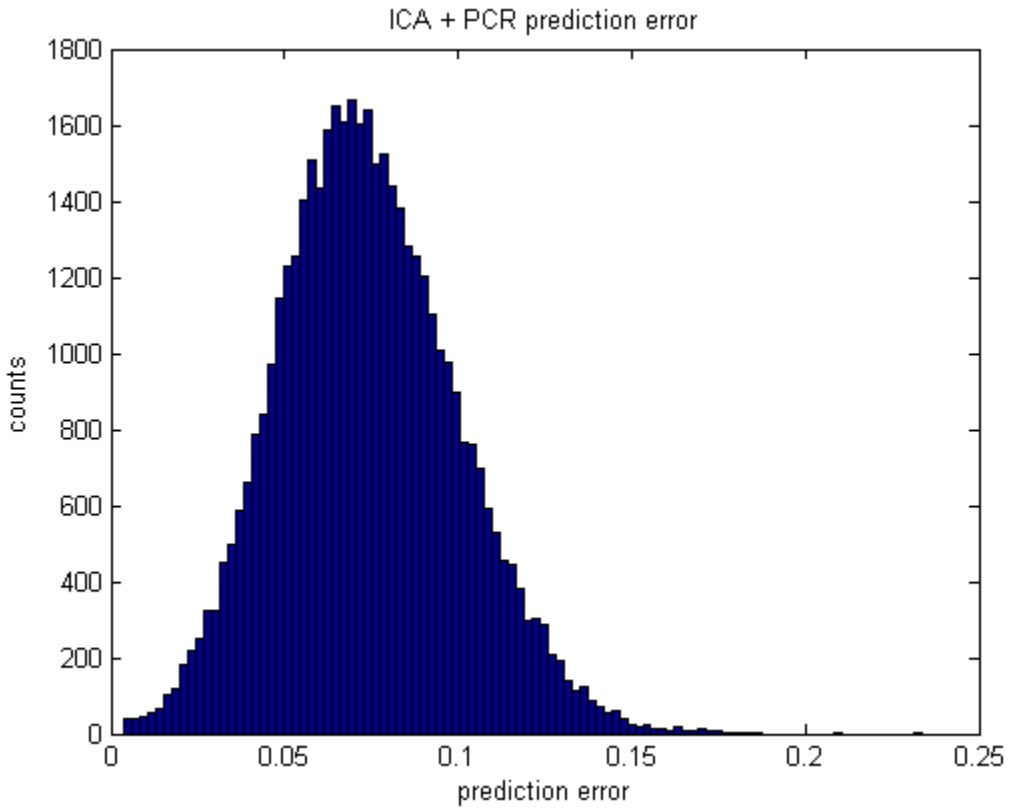

Figure 5.16 Histogram of bootstrap prediction error for ICA + PCR hybrid model.

The prediction intervals are estimated from calculating the expectation from the prediction error distribution. The ICA plus the PCR hybrid model is 0.078 while the PCR model alone is 0.1324. The uncertainty in PCR model is as much as two times that of the hybrid model.

5.7 Summary

In this chapter, a method for ICA uncertainty analysis is developed and applied to actual plant data. The prediction error comes from the bias and variance of model

prediction. A linear model example is used to show that the bootstrap method produces similar prediction interval estimates when compared to standard analytical methods. For the ICA-based data transformation method, no analytic method exists, but the bootstrap method can be successfully used to estimate the prediction interval.

In the example data set, the major contribution to prediction error is from the bias term. ICA produces a low variance prediction. The prediction error is about 0.08% of the total measurement, which provides significant room for sensor drift before mandatory calibration.

The prediction interval of the hybrid model is compared with the PCR model via the bootstrap method. The PCR model produces prediction interval that is twice the size of the hybrid model.

CHAPTER 6
CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

6.1 Conclusions

In this dissertation a novel approach to redundant sensor validation was developed that employs Independent Component Analysis (ICA). Performance of the ICA algorithm indicated sufficiently low parameter estimate residual variances when compared to simple averaging, the Instrument Calibration and Monitoring Program (ICMP), and Principal Component Analysis (PCA) techniques. The drift detection capabilities of the reviewed algorithms were similar for cases in which the number of redundant channels was high. However, for smaller redundant sets down to only 2 channels, only ICA was able to consistently, and correctly, identify the drifting sensor while minimizing the effect of the drifting instrument channel measurements on the parameter estimate. For stationary signals it can detect and isolate sensor drifts for as few as two redundant sensors.

The ICA method is fast and can be embedded into a real-time system. The ICA estimate contains less noise than the original signals and is drift free. Thus the ICA estimate for system control was investigated. Experimental results show that the ICA controller can be used for real time control and can eliminate potential drifts for steady state operations. The usage and effectiveness of ICA was demonstrated in several water level control experiments.

The robustness of the technique was also investigated. The ICA based system was proven to be accurate and robust; however, classical ICA algorithms commonly fail when

distributions are multi-modal. This most likely occurs during highly non-stationary transients. However, the unity check technique indicates such a failure and the method can switch to simple average. For linear non-stationary signals, a rotation transform can resolve the problem, but the implementation is fairly complex.

The ICA technique is easy to apply. Only one parameter needs to be pre-determined: the window size of the data. It is dependent on the sample rate and the necessary speed of response, but should be at least 50. Fault isolation is performed with a simple residual check using tolerances in a control chart framework.

ICA is an unsupervised learning paradigm which builds and applies the model at the same time. Merging ICA with an inferential sensor technique such as PCR provides a prediction is more robust. It also has filtered data anomalies that occur in inferential sensor prediction.

The hybrid system reduces uncertainties in the parameter prediction which can be evaluated using bootstrap prediction intervals. Both the bias and variance of prediction error can be estimated directly from the residuals.

6.2 Future Work

The Classical ICA method failed when it was applied to nonstationary signals. This limited the effectiveness of the method for general usage of signal processing. Though the unity check based reliability module can indicate when the method fails, a direct solution is preferred. The rotation transform solves a linear nonstationary signal problem and also implies a piece- wise solution for more general conditions. However, other nonlinear transforms such as kernel regression might be found to provide a more

general solution. Other ICA principles such as a change of variances may be used to solve some specific classes of nonstationary signal problems.

ICA for sensor validation is based on residual analysis. The more sophisticated scheme in a parity space approach can be utilized in residual analysis. ICA estimation is merely a residual generating engine. This is extremely useful when merging ICA with inferential modeling for common mode failure detection. This approach will expand the scope of sensor validation to general fault detection and isolation problems.

A more sensitive error detection method, such as the SPRT, might be more optimal than a control chart method. However, the distribution assumption in SPRT and a higher rate of false alarms requires future research.

REFERENCES

Amand, Th., G. Heyen, and B. Kalitventzeff, (2001), "Plant monitoring and fault detection synergy between data reconciliation and principal component analysis," *Computers and Chemical Engineering*, **25**, 501-507.

Attias, H. (1999), "Independent factor analysis", *Neural Computation,* **11**(5), 803-852.

Back, A.D., and A.S. Weigend, (1997), "A first application of independent component analysis to extracting structure from stock returns.", *Int. J. on Neural Systems,* 8(4), 473-484.

Basseville, Michèle, (1997), "Information criteria for residual generation and fault detection and isolation," *Automatica*, **33**, no. 5, 783-803.

Benkhedda, H., and R.J. Patton, (1996), "B-spline network integrated qualitative and quantitative fault detection," *Proceedings of IFAC 13th world congress*, June 30-July 5, San Francisco, **N**, 163-168.

Burel, G. (1992), "A nonlinear neural algorithm", *Neural Networks,* **5**, 937-947.

Chan, C.W., J. Hong, K.C. Cheung, and H.Y. Zhang, (1998), "State estimation with measurement error compensation using neural networks," *Proceedings of the IEEE Conference on Control Applications*, Trieste, Italy, September 1-4, 153-157.

Chan, C.W., Hong Jin, K.C. Cheung, H.Y. Zhang, (2001), "Fault detection of systems with redundant sensors using constrained Kohonen networks," *Automatica*, **37**, 1671-1676.

Chowdbury, B.H., and K.Y. Wang, (1996), "Fault classification in power systems using artificial neural networks," *International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications*, **4**, no. 2, 101-112.

Cichocki, A., S. Amari, K. Siwek et al., (2002) ICALAB Toolboxes, http://www.bsp.brain.riken.go.jp/ICALAB.

Comon, P. (1994). Independent component analysis, a new concept? *Signal Processing,* 36, 287-314.

Cook, D., A. Buja, and J. Cabrera, (1993), "Projection pursuit indexes based on orthonormal function expansions.", *J. of Computational and Graphical Statistics,* 2(3), 225-250.

Cover, T. and Thomas, J.A. (1991), Elements of Information Theory, *Wiley Series in Telecommunications.*

Davis, Eddie and Daniel L. Funk, (1995) "Monte Carlo Simulation and Uncertainty Analysis of the Instrument Calibration and Monitoring Program," EPRI WO3785-02.

Deckert, J.C., J. L. Fisher, D.B. Laning, and A. Ray, (1983), "Signal validation for nuclear power plants," *ASME Journal of Dynamic Systems, Measurement, and Control*, 24-29.

Deco, Gustavo and D. Obradovic, (1996), *An Information-Theoretic Approach to Neural Computing,* Springer-Verlag, New York.

Desai, Mukund N., James C. Deckert, and John J. Deyst Jr., (1979), "Dual-sensor failure identification using analytical redundancy," *Journal of Guidance and Control*, **2**, no. 3, 213-220.

Ding, Jun, Andrei Gribok, J. Wesley Hines, Brandon Rasmussen,(2004), "Redundant Sensor Calibration Monitoring Using ICA and PCA", *Real Time Systems* Special Issue on "Applications of Intelligent Real-Time Systems for Nuclear Engineering", Vol. 27(1), 27-48, May 2004.

Ding, Jun, J. Wesley Hines, Brandon Rasmussen, (2003a),"Independent Component Analysis for Redundant Sensor Validation", Proceedings of the 2003 Maintenance and Reliability Conference, Knoxville, TN, May 4-7.

Ding, Jun, J. W. Hines, B. Rasmussen, (2003b), "ICA Filter for Redundant Sensor Monitoring", *Transactions of the American Nuclear Society*, New Orleans, LA, November.

Doymaz, Fuat, Jose A. Romagtnoli, and Ahmet Palazoglu, (2001a), "A strategy for detection and isolation of sensor failures and process upsets," *Chemometrics and Intelligent Laboratory Systems*, **55**, 109-123.

Doymaz, Fuat, James Chen, Jose A. Romagtnoli, and Ahmet Palazoglu, (2001b), "A robust strategy for real-time process monitoring," *Journal of Process Control*, **11**, 343-359.

Dunia, Ricardo, and S. Joe Qin, (1998a), "Joint diagnosis of process and sensor faults using principal component analysis," *Control Engineering Practice*, **6**, 457-469.

Dunia, Ricardo, and S. Joe Qin, (1998b), "A unified geometric approach to process and sensor fault identification and reconstruction: the unidimensional fault case," *Computers and Chemical Engineering*, **22**, no. 7-8, 927-943.

Efron, Bradley and Robert J. Tibshirani (1993), "An Introduction to the Bootstrap", Chapman & Hall, New York.

Efron, B. (1979a) Bootstrap methods: another look at the jackknife. *Ann. Statist.* **7**, 1-26.

Efron, B. (1979b) Computers and the theory of statistics: thinking the unthinkable. *SIAM Review* **21**, 460-480.

Efron, B. (1981) Nonparametric standard errors and confidence intervals. *Can. J. Statist.* **9**, 139-172.

Efron, B. (1987) Better Bootstrap confidence intervals. *J. Amer. Statist. Assoc.* **82**, 171-200.

EPRI, (1981), "On-line power plant signal validation technique utilizing parity-space representation and analytic redundancy", NP-2110, Palo Alto, CA

EPRI, (2000),"On-Line Monitoring of Instrument Channel Performance", Topical Report #104965, Palo Alto, CA.

Farid, H. and E. H. Adelson, (1999), "Separating reflections from images by use of independent component analysis.", *J. of the Optical Society of America,* 16(9), 2136-2145.

Frank, P.M., and X. Ding, (1997), "Survey of robust residual generation and evaluation methods in observer-based fault detection systems," *Journal of Process Control*, **7**, no. 6, 403-424.

Frank, P.M., (1990), "Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy – a survey and some new results," *Automatica*, **26**, no. 3, 459-474.

Friedman, J. H. and J. W. Tukey, (1974), "A projection pursuit algorithm for exploratory data analysis.", *IEEE Trans. of Computers,* c-23(9), 881-890.

Friedman, J. H., (1987), "Exploratory projection pursuit", *J. of the American Statistical Association,* 82(397), 249-266.

Frisk, Erik, and Mattias Nyberg, (2001), "A minimal polynomial basis solution to residual generation for fault diagnosis in linear systems," *Automatica*, **27**, 1417-1424.

Gertler, Janos J., and Ramin Monajemy, (1995), "Generating directional residuals with dynamic parity relations," *Automatica*, **31**, no. 4, 627-635.

Gertler, J., (1997), "Fault detection and isolation using parity relations," *Control Engineering Practice*, **5**, no. 5, 653-661.

Gertler, J.W., W. Li, Y. Huang, and T.J. McAvoy, (1999), "Isolation enhanced principal component analysis," *AIChE Journal*, **45**, no. 2, 323-334.

Giannakopoulos, X., J. Karhunen, and E. Oja, (1999), "Experimental comparison of neural algorithms for independent component analysis and blind separation. *Int. J. of Neural Systems*, 9(2):651-656.

Girolami, Mark, (1999), "Self-Organising Neural Networks, Independent Component Analysis and Blind Source Separation", Springer.

Gribok, A.V., J. Wesley Hines, and Robert E. Uhrig, (2000a), "Use of Kernal Based Techniques for Sensor Validation in Nuclear Power Plants", The *Third American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation and Control and Human-Machine Interface Technologies*, Washington DC, November 13-17.

Gribok, A. V., Ibrahim K. Attieh, J.Wesley Hines, and Robert E. Uhrig, (2000b), "Stochastic Regularization of Feedwater Flow Rate Evaluation for Venturi the Meter Fouling Problem in Nuclear Power Plants", *Inverse Problems in Engineering*.

Gribok, A. V., J. Wesely Hines, Ibrahim Attieh, and Robert E. Uhrig, (2001), "Regularization of Feedwater Flow Rate Evaluation for the Venturi Meter Fouling Problems in Nuclear Power Plants", *Nuclear Technology,*Vol. 134, No. 1, April.

Gribok, A.V., J.W. Hines, A. Urmanov and R.E. Uhrig,(2002), "Heuristic, Systematic, and Informational Regularization for Process Monitoring", by *International Journal of Intelligent Systems,* special issue on Intelligent Systems for Process Monitoring, Vol. 17 No. 8, pp. 723-750, Wiley Publishers.

Gross, K. S. and K. E. Humenik, (1991), "Sequential Probability Ratio Test for Nuclear Plant Component Surveillance", *Nuclear Technology*, Vol. 93, Feb. 1991, pp.131-137.

Gross, K.C., R. M. Singer, S. W. Wegerich, J. P. Herzog, R. Vanalstine and F. Bockhorst (1997), "Application of a Model-based Fault Detection System to Nuclear Plant Signals", Proceedings, Intelligent System Applications to Power Systems, Seoul, Korea, July 6-10, pp. 66-70.

Hall, S.R., P. Motyka, E. Gai, and J.J. Deyst, (1983), "In-flight parity vector compensation for FDI," *IEEE Transactions on Aerospace and Electronic Systems*, AES-19, **5**, 459-474.

Ham, F. M. and N.A. Faour, (1999), "Infrasound signal separation using independent component analysis", *Proc. 21st Seismic Research Symposium: Technologies for Monitoring the Comprehensive Nuclear-Test-Ban Treaty,* Las Vegas, Nevada.

Harman, H. H., "Modern Factor Analysis", University of Chicago Press, 2nd edition, 1967.

Hastie, T., R. Tibshirani, and J. Friedman (2001), "The Elements of Statistical Learning, Data Mining, Inference and Prediction", Springer-Verlag, New York.

Hines, J. W., Darryl J. Wrest, and Robert E. Uhrig, (1996), "Plant Wide Sensor Calibration Monitoring", proceedings of *The 1996 IEEE International Symposium on Intelligent Control*, Dearborn, MI, Sept 15-18.

Hines, J.W., Andrei V. Gribok, Ibrahim Attieh, and Robert E Uhrig, Ed. Da Ruan, (1999), "Regularization Methods for Inferential Sensing in Nuclear Power Plants", *Fuzzy Systems and Soft Computing in Nuclear Engineering*, Springer.

Hines, J. W., Andrei V. Gribok, Robert Uhrig, Ibrahim Attieh, (2000), "Neural Network Regularization Techniques for a Sensor Validation System", *American Nuclear Society Annual Meeting*, San Diego, California, June 4-8.

Hines, J. W. and Brandon Rasmussen,(2001),"On-Line Sensor Calibration Verification: "A Survey"", by Published Proceedings of the *14th International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management*, Manchester, England, Elsevier Publisher, September.

Hines, J.W., Gribok, A.V, Rasmussen, B., J. Ding, (2003),"Redundant Sensor Calibration Monitoring and Reduction System: Final Report", for the Electric Power Research Institute.

Hines, J.W., J. Ding, "ICA Based Redundant Sensor Calibration Monitoring and Reduction System", (2004), for the Electric Power Research Institute.

Huang, Yunbing, Janos Gertler, and Thomas J. McAvoy, (2000), "Sensor and actuator fault isolation by structured partial PCA with nonlinear extensions," *Journal of Process Control*, **10**, 459-469.

Huber, P.J., (1985), "Projection pursuit", *The Annals of Statistics,* 13(2), 435-475.

Hyvarinen, A., (1999a), “Fast and Robust Fixed-Point Algorithms for Independent Component Analysis”, *IEEE Transactions on Neural Networks*, Vol. 10, No. 3.

Hyvarinen, A., (1999b), “Survey on Independent Component Analysis”, *Neural Computing Surveys*, Vol. 2, pp. 94-128.

Hyvarinen, A., J. Karhunen, and E. Oja, (2001) Independent Component Analysis, pp 1-11 & 125-137.

Ikonomopoulos, A., R. E. Uhrig, and L. H. Tsoukalas (1992), "A Methodology for Performing Virtual Measurements in a Nuclear Reactor System", *Transactions of the 1992 American Nuclear Society International Conference on Fifty Years of Controlled Nuclear Chain Reaction: Past, Present, and Future*, Chicago, Illinois, 106-107.

Isermann, R., (1997), "Supervision, fault-deteceion, and fault-diagnosis methods – an introduction," *Control Engineering Practice*, **5**, no. 5, 639-652.

Isbell, C. L. and P. Viola, (1999), "Restructuring sparse high-dimensional data for effective retreval. ", *Advances in Neural Information Processing Systems,* vol. 11, MIT press.

Jin, H. and H.Y. Zhang, (1997), "Configuration of redundant sensor systems and its fault detection using parity vector method," *Proceedings of the IFAC Symposium on fault detection, supervision and safety for technical processes*, August 26-28, Kingston-Upon-Hull, UK, **2**, 843-848.

Joliffe, I.T., (1986), *Principal Component Analysis*, Springer-Verlag, New York.
Jones, M.C. and R. Sibson, (1987), "What is projection pursuit?", *J. of the Royal Statistical Society,* ser. A, 150, 1-36.

Jutten, C. and J. Herault, (1991), "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture.", *Signal Processing,* 24, 1-10.

Kano, Manabu, Koji Nagao, Shinji Hasebe, Iori Hashimoto, Hiromu Ohno, Ramon Strauss, and Bhavik Bakshi, (2000), "Comparison of statistical process monitoring methods: application to the Eastman challenge problem," *Computers and Chemical Engineering*, **24**, 175-181.

Kano, Manabu, Shinji Hasebe, Iori Hashimoto, and Hiromu Ohno, (2001), "A new multivariate statistical process monitoring method using principal component analysis," *Computers and Chemical Engineering*, **25** 1103-1113.

Kendall, M., (1975), "Multivariate Analysis", Charles Griffin & Co.

Kerr, T.H., (1983), "The controversy over the use of SPRT and GLR techniques and other loose ends in failure detection," *Proc. of the American Control Conference*, June, 966-977.

Krishnaswami, V., G.-C. Luh, and G. Rizzoni, (1995), "Nonlinear parity equation based residual generation for diagnosis of automotive engine faults," *Control Engineering Practice*, **3**, no. 10, 1385-1392.

Lee, T.W., B. Koehler and R. Orglmeister (1997), "Blind separation of nonlinear mixing models", *IEEE NNSP,* 406-415.

Lee, T.W., (1998), "Independent Component Analysis, Theory and Applications", Kluwer Academic Publishers.

Lewicki, M. and T.J. Sejnowski (2000), "Learning over complete representations", *Neural Computation,* **12**(2), 337-365.

Matsuoka, K., M. Ohya and M. Kawamoto, (1995), "A neural net for blind separation of nonstationary signals", *Neural Networks,* **8**(3), 411-419.

Marbach, R., and H.M. Heise, (1990), "Calibration modeling by partial least-squares and principal component regression and its optimization using an improved leverage correction for prediction testing," *Chemometrics and Intelligent Laboratory Systems*, **9**, 45-63.

Martens, H., and T. Naes, (1989), *Multivariate Calibration*, John Wiley & Sons, Chichester.

Molgedey, L., and H. G. Schuster, (1994), "Separation of a mixture of independent signals using time delayed correlations.", *Physical Review Letters,* 72, 3634-3636.

Murata, N., K.R. Mueller, A. Ziehe, and S. Amari (1997), "Adaptive on-line learning in changing environments", In *Advances in Neural Information Processing Systems 9*, 599-605, MIT Press.

Nadal, J. P. and N. Parga, (1994), "Non-linear neurons in the low noise limit: a factorial code maximizes information transfer", *Network*, **5**, 565-581.

Nuzillard, D. and J. –M. Nuzillard, (1998), "Application of blind source separation to 1-d and 2-d nuclear magnetic resonance spectroscopy.", *IEEE Signal Processing Letters,* 5(8), 209-211.

Nyberg, Mattias, (1997), "Parity functions as universal residual generators and tool for fault detectability analysis," *Proceedings of the 36th IEEE Conference on Decision and Control*, **5**, 4483-4489.

Parra, Lucas and Clay Spence,(2000), "Convolutive Blind Separation of Non-Stationary Sources", IEEE Transactions on Speech and Audio Processing, Vol. 8 No. 3, May.

Patton, R. J., and J. Chen, (1991), "A review of parity space approaches to fault diagnosis," *Proc. IFAC Symp. SAFEPROCESS'91*, Baden-Baden, September 10-13.

Patton, R.J., and J. Chen, (1997), "Observer-based fault detection and isolation: robustness and applications," *Control and Engineering Practice*, **5**, no. 5, 671-682.

Qin, S. J. and T. J. McAvoy (1992), "Nonlinear PLS Modeling Using Neural Networks", *Computers in Chemical Engineering,* vol. 16, no.4, pp. 379-391.

Qin, S. J. and W. Li (1999), "Detection, identification and reconstruction of faulty sensors with maximized sensitivity", *AIChE J.,* 45(9).

Rasmussen, Brandon, J. Wesley Hines, and Robert E. Uhrig, (2000), "Nonlinear Partial Least Squares Modeling for Instrument Surveillance and Calibration Verification", proceedings of the *Maintenance and Reliability Conference (MARCON 2000)*, Knoxville, TN, May 7-10.

Rasmussen, B, A.V. Gribok, J.W. Hines, "Redundant Sensor Calibration Monitoring and Reduction System Project". Progress Report for the Electric Power Research Institut, July 18, 2002.

Rasmussen, Brandon, (2003), "Prediction Interval Estimation Techniques for Empirical Modeling Strategies and their Applications to Signal Validation Tasks", Ph.D Dissertation, University of Tennessee.

Ray, Asok, and Shashi Phoha, (2002), "Detection and identification of potential faults via multi-level hypotheses testing," *Signal Processing*, In Press.

Ray, Asok, and Shashi Phoha, (2000), "Calibration and estimation of redundant signals," *Automatica*, **36**, 1525-1534.

Ray, Asok, and R. Luck, (1991), "Signal validation in multiply-redundant systems," *IEEE Control Systems Magazine*," **11**, no. 2, 44-49.

Ray, Asok, (1989), "Sequential testing for fault detection in multiply-redundant systems," *Journal of Dynamic Systems, Measurement, and Control*, **111**, 329-332.

Ray, Asok, and Mukund Desai, (1986), "A redundancy management procedure for fault detection and isolation," *Journal of Dynamic Systems, Measurement, and Control*, **108**, 248-254.

Ray, A., and M. Desai, (1984), "A calibration and estimation filter for multiply redundant measurement systems," *ASME Journal of Dynamic Systems, Measurement, and Control*, 149-156.

Ray, Asok, Robert Geiger, Mukund Desai, and Kohn Deyst, (1983a), "Analytic redundancy for on-line fault diagnosis in a nuclear reactor," *Journal of Energy*, **7**, no. 4, 367-373.

Ray, Asok, Mukund Desai, and Kohn Deyst, (1983b), "On-line fault diagnosis in a nuclear reactor by sequential testing," *IEEE Trans. Nuc. Sci.*, **NS-30**, 1850-1855.

Roberts, Stephen and R. Everson (2001), "Independent Component Analysis, Principles and Practice", Cambridge.

Schimek, Michael G., Editor, (2000), "Smoothing and Regression," John Wiley & Sons, Inc.

Seiter, J.C., and M.D. DeGrandpre, (2001), "Redundant chemical sensors for calibration-impossible applications," *Talanta*, **54**, 99-106.

Shannon, C.E. (1948) "A Mathematical Theory of Communication". *Bell Sys. Tech. Journal,* **27**, 379-423.

Simani, S., Ron J. Patton, Steve Daley, and Andrew Pike, (2000), "Identification and fault diagnosis of an industrial gas turbine prototype model," *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, December, 2615-2620.

Spayer, J.L., and J.E. White, (1984), "The Shiryaev sequential probability ratio test for redundancy management," *Jounral of Guidance, Control, and Dynamics*, **7**, no. 5, 588-595.

Sun, J., (1993), "Some practical aspects of exploratory projection pursuit", *SIAM J. of Sci. Comput.,* 14, 68-80.

Taleb, A. and C. Jutten, (1997), "Nonlinear source separation: the post-nonlinear mixtures", *ESANN,* 279-284.

Tibshirani, Robert, (1996), "A comparison of some error estimates for neural network models", *Neural Computation,* **8**, 152-163.

Torkkola, K., (1999), "Blind separation for audio signals – are we there yet?", *Proc. Int. Workshop on Independent Component Analysis and Signal Seperation (ICA '99)*, 239-244, Aussois, France.

Uhrig, R.E., Darryl J. Wrest and J. Wesley Hines, (1997), "Intelligent Surveillance and Calibration Verification in Power Systems", proceedings of *Intelligent System Applications to Power Systems*, Korea, July 6-10.

Uhrig, Robert E., Lefteri H. Tsoukalas, (1999), "Soft Computing Technologies in Nuclear Engineering Applications", *Progress in Nuclear Energy,* Vol. 34, No. 1, pp 13-75.

Upadhyaya, B.R., (1985), "Sensor Failure Detection and Estimation", *Nuclear Safety.*

Upadhyaya, B. R. and E. Eryurek, (1992), "Application of Neural Networks for Sensor Validation and Plant Monitoring", *Nuclear Technology*, Vol. 97, 170-176.

Urmanov, A., Andrei Gribok, J.W. Hines, and Brandon Rasmussen, (2000), "Application of Information Complexity in Principal Component Regression Modeling of the Venturi Meter Drift", proceedings of the *Maintenance and Reliability Conference (MARCON 2000)*, Knoxville, TN, May 6-9.

Vapnik, Vladimir N. (1998), *Statistical Learning Theory,* pp 1 – 58.

Venkateswaran, N., M.S. Siva, and P.S. Goel, (2002), "Analytical redundancy based fault detection of gyroscopes in spacecraft applications," *Acta Astronautica*, **50**, no. 9, 535-545.

Vigneau, E., E.M. Qannari, and D. Bertrand, (2002), "A new method of regression on latent variables. Application to spectral data," *Chemometrics and Intelligent Laboratory Systems*.

Wald, A., (1947), *Sequential Analysis*, John Wiley, New York.

Willsky, Alan S., (1976), "A survey of design methods for failure detection in dynamic systems," *Automatica*, **12**, 601-611.

Wolpert, D. (1992), "On the connection between in-sample testing and generalization error", *Complex Systems,* **6**, 47-94.

Wolpert, D. (1994), "The relationship between PAC, the Statistical Physics framework, the Bayesian framework, and the VC framework", *The Mathematics of Generalization,* Addison-Wesley.

Wolpert, D., (1995a), "On the Bayesian 'Occam factors' argument for Occam's razor", *Computational Learning Theory and Natural Learning Systems: Volume III,* MIT Press.

Wolpert, D. and Macready, W. (1995b), "No Free Lunch Theorems for Search", SFI TR 95-02-010.

Wooten, B., (1993) "Instrument Calibration and Monitoring Program Volume 1: Basis for the Method," EPRI TR-103436-V1.

Xu, Xiao and J. Wesley Hines, (1998), "On-Line Sensor Calibration Monitoring and Fault Detection for Chemical Processes", proceedings of the *Maintenance and Reliability Conference (MARCON 98)*, Knoxville, TN, May 12-14.

Xiao Xu, J. Wesley Hines, Robert E. Uhrig, (1999), "Sensor Validation and Fault Detection Using Neural Networks", proceedings of the *Maintenance and Reliability Conference (MARCON 99)*, Gatlinburg, TN, May 10-12.

Yang, H., S. Amari and A. Cichocki (1997), "Information back-propagation for blind separation of sources from non-linear mixtures", *Proc. of ICNN,* 2141-2146, Houston.

Ypma A. and P. Pajunen, (1999), "Rotating machine vibration analysis with second-order independent component analysis.", *Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA '99),* 37-42, Aussois, France.

Appendix

A.1 Published Articles

Five articles have been published during the development of this research. They are briefly described below.

A.1.1 “Redundant Sensor Calibration Monitoring Using ICA and PCA"

This paper was published in *Real Time Systems* Special Issue on "Applications of Intelligent Real-Time Systems for Nuclear Engineering", Vol. 27(1), 27-48, May 2004. This paper presents a comparison of methods for industrial on-line sensor calibration monitoring for redundant sensors. Principal component analysis (PCA) and independent component analysis (ICA) techniques are developed and compared using both simulated data and data sets from an operating nuclear power plant. The performance is dependent on the types of noise sources; however, under most conditions ICA outperforms PCA, based on the bias and variance of their respective parameter estimates. A case study is included to demonstrate the usefulness of both techniques for the early detection of sensor drift.

A.1.2 "Independent Component Analysis for Redundant Sensor Validation"

This paper was published in the Proceedings of the 2003 Maintenance and Reliability Conference, Knoxville, TN, May 4-7. This paper mainly compared ICA method and ICMP results. The ICA method is able to reduce the redundancy of the original dataset in order to predict the process parameter more accurately. The ICA prediction method is proven to be a robust method that can be used as a non-parametric approach to build a model that can detect faulty and drifted sensors so that they can be scheduled for

maintenance. A slow sensor drift case study from a nuclear power plant is presented to show the usefulness of this technique. The ICA based system results are much better than other current methods such as ICMP. Independent component analysis is shown to be a new and effective approach for redundant sensor validation.

A.1.3 "ICA Filter for Redundant Sensor Monitoring"

This paper was published in the *Transactions of the American Nuclear Society*, New Orleans, LA, November, 2003. In this paper, the noise reduction feature of ICA is presented. ICA estimate contains less noise and is drift-free. It acts as a noise and drift filter to the redundant measurements. Additional FDI technique is no longer needed while applying ICA filtering. In other words, ICA filtering itself acted as FDI technique. The measurements from each channel contain the process parameter, a common noise source and independent channel noise sources. These three components are most likely independent from each other. Except for the channel noise, the other two components are seldom a Gaussian distribution. Another assumption for ICA is that the transform matrix **A** is linear and time invariant. These assumptions are valid at most conditions, but are especially valid for fairly steady state measurements. Moreover, during faulty conditions, the fault component is introduced into one or more redundant channels and is absolutely independent from the process parameter. Therefore, we can create the model not only from fault free data (regression), but we can also build the model using data when the drift is present because of the model's ability to separate the independent components. ICA filtering is applied on a three-channel measurement which one of the channel is

drifting. The original measurements and the drift-free ICA estimate are shown in the article.

A.1.4 "A Robust Controller for Two Redundant Sensor Systems"

This paper was published in the Proceedings of the 2004 Maintenance and Reliability Conference, Knoxville, TN, May 4-7. Redundant sensors are often used to measure critical process variables that are controlled. When double redundancy is used, one must choose which of the two sensors to use for input to the controller. If a drifting or faulty sensor is chosen to be the controller input, the system will be incorrectly operated. Independent component analysis (ICA) provides a solution to this problem. ICA prediction is robust in that the faulty component of a sensor measurement does not adversely affect the parameter estimate; even there are only two redundant sensors. A new control strategy is presented that uses the ICA estimate as the controller input, so that drift in one of the sensors will not cause the controlled system to drift. The ICA-based control strategy is demonstrated on a water level control experiment. The ICA controller is compared to a classical controller during normal and drifting conditions. The results demonstrate that in the event of sensor drifts, the system is correctly controlled and the drifting sensor is correctly identified.

A.1.5 "A Hybrid Redundant Sensor Estimation Technique for 2-channel Systems"

This paper will be published at the 14th Annual Joint ISA POWID/EPRI Controls and Instrumentation Conference and 47th Annual ISA Power Industry Division Symposium, Colorado Springs, Colorado, June 6-11. A Redundant Sensor Estimation Technique

(RSET) was developed for on-line redundant sensor calibration monitoring. This method has been extended to the two-channel case through the use of an inferential sensor. The inferential sensor uses an empirical model with correlated signals as inputs. The advantages of this system are reduced spillover and noise characteristics through the use of the Independent Component Analysis based RSET system and increased stability due to the inferential sensor. Theses advantages are demonstrated through an application in which first stage turbine pressure sensors of a nuclear power plant are monitored.

A.2 Matlab Functions

A.2.1 ICA Interface

```
% Redundant Sensor Calibration Monitoring and Reduction System
% RSET
% Rset_gui.m User interface for RSET
% Written by Jun Ding Version 1.0
% Jan. 3, 2003
% University of Tennessee
% Copyright @ 2003 All rights reserved


global fname;
global sName;
global sUnits;
global sTag;
global sSpan;
global sPTID;
global sGroupID;
global inputX;
varPlot=10;

set(gcf,'DefaultUicontrolUnits','Normalized')
color_order=['b' 'g' 'r' 'c' 'm' 'y' 'k' 'b' 'g' 'r' 'c' 'm' 'y' 'k'];

%Frame Input
text_title_ = uicontrol(gcf,'Style','Text',...
   'String','Redundant Sensor Calibration Monitoring and Reduction System',...
   'Position',[0.1 0.95 0.8 0.05],'HorizontalAlignment','Center');
```

```
set(text_title_,'Fontname','Palladius','FontSize',0.03,'FontWeight','light');
frame1_ = uicontrol(gcf,'Style','Frame','Position',[0.054 0.678 0.395 0.27]);
text_f1_ = uicontrol(gcf,'Style','Text',...
   'String','Input Signals',...
   'Position',[0.163 0.90 0.165 0.03],'FontWeight','bold','HorizontalAlignment','Center');
push_load_ = uicontrol(gcf,'Style','Pushbutton','String','Load','Value',0,'Position',[0.067
0.818 0.072 0.05],'Callback','getFile');
edit_f_ = uicontrol(gcf,'Style','Edit','String',fname,'Position',[0.196 0.818 0.162
0.05],'HorizontalAlignment','left');
text_f2_ = uicontrol(gcf,'Style','Text',...
   'String','Redundancy',...
   'Position',[0.067 0.758 0.125 0.035],'HorizontalAlignment','Left');
edit_f2_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.196 0.758 0.10
0.05],'HorizontalAlignment','left');
text_i3_ = uicontrol(gcf,'Style','Text',...
   'String','Sampling Rate',...
   'Position',[0.067 0.70 0.185 0.035],'HorizontalAlignment','Left');
edit_i3_ = uicontrol(gcf,'Style','Edit','String','1','Position',[0.196 0.70 0.10
0.05],'HorizontalAlignment','left');
push_transpose_ =
uicontrol(gcf,'Style','Pushbutton','String','Transpose','Value',0,'Position',[0.307 0.758 0.10
0.05],'Callback','transVar');
push_plot_ = uicontrol(gcf,'Style','Pushbutton','String','Plot','Value',0,'Position',[0.307
0.70 0.072 0.05],'Callback','plotVar');

%Frame Description
frame2_ = uicontrol(gcf,'Style','Frame','Position',[0.492 0.678 0.395 0.27]);
text_f3_ = uicontrol(gcf,'Style','Text',...
   'String','Description of Sensors',...
   'Position',[0.56 0.91 0.265 0.03],'FontWeight','bold','HorizontalAlignment','Center');
text_s1_ = uicontrol(gcf,'Style','Text',...
   'String','Name',...
   'Position',[0.502 0.85 0.05 0.03],'HorizontalAlignment','Left');
text_s2_ = uicontrol(gcf,'Style','Text',...
   'String','Units',...
   'Position',[0.502 0.776 0.05 0.03],'HorizontalAlignment','Left');
text_s3_ = uicontrol(gcf,'Style','Text',...
   'String','Tag#',...
   'Position',[0.502 0.702 0.05 0.03],'HorizontalAlignment','Left');
edit_s1_ = uicontrol(gcf,'Style','Edit','String',sName,'Position',[0.562 0.85 0.12
0.05],'HorizontalAlignment','left');
edit_s2_ = uicontrol(gcf,'Style','Edit','String',sUnits,'Position',[0.562 0.776 0.12
0.05],'HorizontalAlignment','left');
edit_s3_ = uicontrol(gcf,'Style','Edit','String',sTag,'Position',[0.562 0.702 0.12
0.05],'HorizontalAlignment','left');
```

```
text_s4_ = uicontrol(gcf,'Style','Text',...
   'String','Span',...
   'Position',[0.692 0.85 0.08 0.03],'HorizontalAlignment','Left');
text_s5_ = uicontrol(gcf,'Style','Text',...
   'String','Pt ID',...
   'Position',[0.692 0.776 0.08 0.03],'HorizontalAlignment','Left');
text_s6_ = uicontrol(gcf,'Style','Text',...
   'String','Group#',...
   'Position',[0.692 0.702 0.08 0.03],'HorizontalAlignment','Left');
edit_s4_ = uicontrol(gcf,'Style','Edit','String',sSpan,'Position',[0.772 0.85 0.10
0.05],'HorizontalAlignment','left');
edit_s5_ = uicontrol(gcf,'Style','Edit','String',sPTID,'Position',[0.772 0.776 0.10
0.05],'HorizontalAlignment','left');
edit_s6_ = uicontrol(gcf,'Style','Edit','String',sGroupID,'Position',[0.772 0.702 0.10
0.05],'HorizontalAlignment','left');

%Frame Moving Window
frame31_ = uicontrol(gcf,'Style','Frame','Position',[0.054 0.057 0.395 0.27]);
text_f41_ = uicontrol(gcf,'Style','Text',...
   'String','Moving Window Size',...
   'Position',[0.064 0.284 0.365 0.03],'FontWeight','bold','HorizontalAlignment','Center');
text_c31_ = uicontrol(gcf,'Style','Text',...
   'String','Start at',...
   'Position',[0.064 0.182 0.1 0.03],'HorizontalAlignment','Left');
text_c32_ = uicontrol(gcf,'Style','Text',...
   'String','End  at',...
   'Position',[0.064 0.132 0.1 0.03],'HorizontalAlignment','Left');
edit_f3_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.164 0.182 0.122
0.05],'HorizontalAlignment','left');
edit_f4_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.164 0.132 0.122
0.05],'HorizontalAlignment','left',...
   'CallBack','setWindow');
set(edit_f3_,'backgroundColor',[1 1 1],'FontSize',0.03);
set(edit_f4_,'backgroundColor',[1 1 1],'FontSize',0.03);
push_plot2_ = uicontrol(gcf,'Style','Pushbutton','String','Plot','Value',0,'Position',[0.322
0.152 0.072 0.05],'Callback','plotWindowVar');

%Frame Drift Injection
frame32_ = uicontrol(gcf,'Style','Frame','Position',[0.054 0.375 0.395 0.27]);
text_f42_ = uicontrol(gcf,'Style','Text',...
   'String','Drift Injection',...
   'Position',[0.064 0.603 0.365 0.03],'FontWeight','bold','HorizontalAlignment','Center');

text_d3_ = uicontrol(gcf,'Style','Text',...
   'String','Ch#',...
```

```
   'Position',[0.067 0.549 0.056 0.04],'HorizontalAlignment','Left');
text_d2_ = uicontrol(gcf,'Style','Text',...
   'String','Start',...
   'Position',[0.067 0.491 0.056 0.04],'HorizontalAlignment','Left');
text_d1_ = uicontrol(gcf,'Style','Text',...
   'String','Type',...
   'Position',[0.067 0.424 0.056 0.04],'HorizontalAlignment','Left');
text_d4_ = uicontrol(gcf,'Style','Text',...
   'String','Parameter',...
   'Position',[0.265 0.549 0.09 0.04],'HorizontalAlignment','Left');
text_d5_ = uicontrol(gcf,'Style','Text',...
   'String','End',...
   'Position',[0.265 0.491 0.056 0.04],'HorizontalAlignment','Left');
edit_d1_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.133 0.491 0.099
0.04],'backgroundColor',[1 1 1],'HorizontalAlignment','left');
edit_d2_ = uicontrol(gcf,'Style','Edit','String','1','Position',[0.133 0.549 0.099
0.04],'backgroundColor',[1 1 1],'HorizontalAlignment','left');
edit_d3_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.353 0.549 0.084
0.04],'backgroundColor',[1 1 1],'HorizontalAlignment','left');
edit_d4_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.353 0.491 0.084
0.04],'backgroundColor',[1 1 1],'HorizontalAlignment','left');
popup_d1_ = uicontrol(gcf,'Style','Popupmenu','backgroundColor',[1 1 1],...
   'String','Ramp UP|Ramp DOWN|Step UP|Step DOWN|Constant at Mean|Constant at
Current Value|Constant at Specific Value|Constant Step UP %|Constant Step DOWN
%|Variance Increase','Value',1,...
   'Position',[0.133 0.424 0.15 0.04],'CallBack','diPopup');
push_plotd_ = uicontrol(gcf,'Style','Pushbutton','String','Plot','Value',0,'Position',[0.322
0.388 0.072 0.05],'Callback','plotWindowVar');

%Frame Transform
frame4_ = uicontrol(gcf,'Style','Frame','Position',[0.492 0.375 0.395 0.27]);

text_f5_ = uicontrol(gcf,'Style','Text',...
   'String','Transform',...
   'Position',[0.56 0.60 0.265 0.03],'FontWeight','bold','HorizontalAlignment','Center');
text_t1_ = uicontrol(gcf,'Style','Text',...
   'String','Bootstrapping Cycle',...
   'Position',[0.51 0.518 0.20 0.04],'HorizontalAlignment','Left');
edit_t1_ = uicontrol(gcf,'Style','Edit','String','0','Position',[0.735 0.518 0.099
0.043],'HorizontalAlignment','left',...
   'CallBack','setCycle');
set(edit_t1_,'backgroundColor',[1 1 1],'FontSize',0.02);
push_do_ = uicontrol(gcf,'Style','Pushbutton','String','Do','Value',0,'Position',[0.518 0.423
0.072 0.05],'Callback','doTransform');
```

```
push_show_ = uicontrol(gcf,'Style','Pushbutton','String','Show','Value',0,'Position',[0.643
0.423 0.072 0.05],'Callback','showTransform');
push_save_ = uicontrol(gcf,'Style','Pushbutton','String','Save','Value',0,'Position',[0.77
0.423 0.072 0.05],'Callback','saveTransform');

%Frame Drift Detection
frame5_ = uicontrol(gcf,'Style','Frame','Position',[0.492 0.057 0.395 0.27]);

text_f6_ = uicontrol(gcf,'Style','Text',...
   'String','Drift Detection',...
   'Position',[0.56 0.285 0.265 0.03],'FontWeight','bold','HorizontalAlignment','Center');
text_dd1_ = uicontrol(gcf,'Style','Text',...
   'String','Method',...
   'Position',[0.521 0.191 0.120 0.04],'HorizontalAlignment','Left');
popup_dd1_ = uicontrol(gcf,'Style','Popupmenu','backgroundColor',[1 1 1],...
   'String','Threshold|SPRT','Value',1,...
   'Position',[0.656 0.211 0.15 0.033],...
   'CallBack','ddPopup');
choice=1;
push_dshow_ = uicontrol(gcf,'Style','Pushbutton','String','Show','Value',0,'Position',[0.518
0.102 0.072 0.05],'Callback','showDD');
push_dsave_ = uicontrol(gcf,'Style','Pushbutton','String','Save','Value',0,'Position',[0.782
0.102 0.072 0.05],'Callback','saveDD');
push_quit_ = uicontrol(gcf,'Style','Pushbutton','String','Quit','Value',0,'Position',[0.919
0.006 0.072 0.05],'Callback','fc_quit');

%dipopup.m

if ~isempty(inputX)
   diChoice = get(popup_d1_,'Value');
   bd = get(edit_d1_,'String');
   bd = str2num(bd);
   sig = get(edit_d2_,'String');
   sig = str2num(sig);
   param = get(edit_d3_,'String');
   param = str2num(param);
   ed = get(edit_d4_,'String');
   ed = str2num(ed);
   sr = get(edit_i3_,'String');
   sr = str2num(sr);

   drift_method=diChoice;
   perc_method=1;
   ref=0;
```

```
   if diChoice==7
     ccv=param;
   elseif diChoice==10;
     sigma=param;
   else
     perc=param;
     ccv=1;
     sigma=1;
   end


driftX=simulate_drift(inputX,sig,perc,perc_method,ref,bd,ed,sr,drift_method,ccv,sigma);
   XX=driftX;
end

%doTransform.m

clear WW_c;
% ICA transform
XY=XX';
X=XY;
dim=size(XY);
icNum=min(4,dim(1));
count=0;
for i=1:rsCycle      % resampling cycles
% prepare new data from resampling
   i=i
   for j=1:dim(2)
      k1=round(rand(1)*(dim(2)-1))+1;
      for k=1:dim(1)
         X(k,j)=XY(k,k1);
      end
   end

[Y,A,W]=FASTICA(X,'approach','symm','g','tanh','displayMode','off','verbose','off','num
OfIC',icNum);
   if ~ isempty(Y)
      count=count+1;
      [W1, s_IC,xavg,index] = icapost(X,Y,W);
      for j=1:dim(1)
        WW_c(count,j)=W1(index,j);
      end
   end
end
```

```
function [W1, s_IC,xavg,IC_index] = icapost(X,Y,W)
% function of ICA post processing
% Identify main component of process from unordered independant components
% X: m(sensor)x n (sampling points) matrix of measured signals
% Y: identified independant components
% xavg: simple average
% IC(m) seperated components of Y
% W1: weight matrix such as Y = W1 * X
% s_IC: main components
% written by Jun Ding
% Sepetember 6 2002

m=size(Y);
IC=zeros(m(1),m(2));
t=linspace(1,m(2),m(2));

for i=1:m(1)
  IC(i,:)=Y(i,:);
end

xavg=mean(X,1);
fac1=mean(median(X)); %robust mean estimation using median

for i=1:m(1)
  m_IC=IC(i,:);
  fac_mean=fac1/median(m_IC);

  if fac_mean < 0
     s_IC= - m_IC*abs(fac_mean);
     W1=-W*abs(fac_mean);
  else
     s_IC=m_IC*abs(fac_mean);
     W1=W*abs(fac_mean);
  end

%  ss(i)=std(s_IC)
  ss(i)=std(s_IC-xavg)
end

minstd=min(ss);


for i=1:m(1)
    if ss(i)==minstd
          IC_index=i;
```

```
        m_IC=IC(i,:);
        fac_mean=fac1/median(m_IC);
      if fac_mean < 0
        s_IC= - m_IC*abs(fac_mean);
        W1=-W*abs(fac_mean);
      else
        s_IC=m_IC*abs(fac_mean);
        W1=W*abs(fac_mean);
      end
    end
 end

function
[driftX]=simulate_drift(X,sig,perc,perc_method,ref,bd,ed,sr,drift_method,ccv,sigma);

% Function to simulate drifts in matrices of sensor measurements

% [driftX]=simulate_drift(X,sig,perc,perc_method,ref,bd,ed,sr,drift_method,ccv,sigma);

%X is an nxp data matrix (use raw data), where n is the number of samples and p is the
number of signals
%sig is the column of the signal to drift
%
%perc is the percent drift, or step
%
%if perc_method=0, the inserted drift is applied as % drift per day
%if perc_method=1, the inserted drift begins at point bd, and ends at point ed at which
point the drift reaches the specified percentage
%
% Note that setting perc_method is only necessary for ramp drifts. If inserting any other
type of drift,
% perc_method can be set to 0 or 1 with the same result.
%
%ref, is the value to which the percentage refers:
%   for percent of the mean, set ref=0
%   for percent of the full signal range, set ref=1
%
% Note this option is to handle drift simulations for signals whose mean values
%   are near zero. A 2% drift for a signal of mean value ~0.1 will not be very visible;
%   in this case, apply the percent to the full signal range to get a better simulation.
%
%
% bd is the point at which to begin the drift
% ed is the point at which to end the drift
% IF ed is set to ed=0, the drift will continue through the end of the data file
```

```
%
%sr is the sampling rate in minutes
%
%drift method specifies the type of drift, rampup (1), rampdown (2), stepup(3), stepdown
(4),
% constant at mean value (5), constant at current value of bd (6), constant at specific
value (7),
% constant step up by a percentage of mean (8), constant step down by a percentage of
the mean (9),
% variance increase (10)
%
%ccv is only used for drift_method=7, otherwise it is not used.
%   It is the value at which a constant step will be inserted.
%   i.e. all points from bd:ed will be set to ccv
%
%sigma is only used for drift_method=10
%   It is the standard deviation of the noise to add from N(0,sigma)
%
% EXAMPLE
% Insert a drift in signal 2 (sig=2) beginning at point 100 (bd=100), ending at point 200
(ed=200).
% The drift will reach a total of 2% (perc=2, perc_method=1) of the mean value (ref=0).
The data sampling rate is 1 minute (sr=1).
% The drift will ramp up (drift_method=1). Ignore ccv and sigma, only used for variance
increase and constant values.
% [driftX]=simulate_drift(X,sig,perc,perc_method,ref,bd,ed,sr,drift_method,ccv,sigma);
% Developed by Brandon Rasmussen, 2003




if ed==0;
   ed=size(X,1)+1;
end;

num_points=ed-bd;

if ref==0;
   M=mean(X(:,sig));
elseif ref==1;
   M=max(X(:,sig))-min(X(:,sig));
end;


%Calculate drift fraction per sample
```

```
f=(sr*perc)/(100*24*60);
fv=f*ones(num_points,1);
cfv=cumsum(fv);

f2=perc/(num_points*100);
fv2=f2*ones(num_points,1);

cfv2=cumsum(fv2);


driftX=X;



   if perc_method==0;

      if drift_method==1; %(ramp up)
         driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)+(cfv*M);
      elseif drift_method==2; %(ramp down)
         driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)-(cfv*M);
      end;

   elseif perc_method==1;
      if drift_method==1; %(ramp up)
         driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)+(cfv2*M);
      elseif drift_method==2; %(ramp down)
         driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)-(cfv2*M);
      end;
   end;

   if drift_method==3; %stepup(3)
         driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)+((perc/100)*M);
   elseif drift_method==4; % stepdown (4)
         driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)-((perc/100)*M);
   elseif drift_method==5; % constant at mean value (5)
         driftX(bd:(ed-1),sig)=M;
      elseif drift_method==6; % constant at current value of bd (6)
         driftX(bd:(ed-1),sig)=driftX(bd,sig);
      elseif drift_method==7; % constant at specific value (7)
         driftX(bd:(ed-1),sig)=ccv;
      elseif drift_method==8; %constant step up by a percentage of mean (8)
         driftX(bd:(ed-1),sig)=M+((perc/100)*M);
      elseif drift_method==9; % constant step down by a percentage of the mean (9)
         driftX(bd:(ed-1),sig)=M-((perc/100)*M);
```

```
      elseif drift_method==10; % variance increase (10)
          driftX(bd:(ed-1),sig)=X(bd:(ed-1),sig)+normrnd(0,sigma,length(bd:(ed-1)),1);
   end;

 %ShowTransform.m

nn=size(WW_c);
if (nn(1)==1)
   tr=WW_c
else
   tr=median(WW_c)
end
PE=XX*tr';
figure(varPlot);
xavg=mean(XX');
plot(PE);
hold on;
%plot(xavg,'r-');
ylabel('parameter estimate');
grid;
title('Redundant Sensors');
varPlot=varPlot+1;

%ShowDD.m

dim=size(XX);
nn=size(WW_c);
if (nn(1)==1)
   tr=WW_c;
else
   tr=median(WW_c);
end
PE=XX*tr';
if ~isempty(PE)
   for i=1:dim(1)
      for j=1:dim(2)
         diff(i,j)=XX(i,j)-PE(i,1);
      end
   end
   mm_diff=ones(dim(1),1)*mean(diff(1:100,:));
   z_diff=diff-mm_diff;
   std_diff=std(diff(1:50,:));
   for j=1:dim(2)
      dline(:,j)=std_diff(1,j)*2*ones(dim(1),1);
      ndline(:,j)=-std_diff(1,j)*2*ones(dim(1),1);
```

```
    lline(:,j)=0.015*mean(XX(:,j))*ones(dim(1),1);
  end

  for i=5:dim(1)
    for j=1:dim(2)
      z_diff(i,j)=(z_diff(i-4,j)+z_diff(i-3,j)+z_diff(i-2,j)+z_diff(i-1,j)+z_diff(i,j))/5;
      dline(i,j)=dline(i,j)+z_diff(i,j);
      ndline(i,j)=ndline(i,j)+z_diff(i,j);
    end
  end

  if choice==1
    for j=1:dim(2)
      figure(varPlot);
      ch=num2str(j);
      ch=strcat('ch#',ch);
      plot(z_diff(:,j),color_order(j));
      text(0.9,0.9,ch,'units','normalized','color',color_order(j))
      hold on;
      plot(dline(:,j),'r-');
      plot(ndline(:,j),'r-');
      plot(lline(:,j));
      plot(-lline(:,j));
      ylabel('difference');
      title('Residual');
      varPlot=varPlot+1;
    end
  end
end
clear diff;
clear dline;

%PlotVar.m

x_size=size(inputX);
figure(varPlot);
text_x=0.9;
text_y=0.9;
for i=1:x_size(2)
  plot(inputX(:,i),color_order(i));
  ch=num2str(i);
  ch=strcat('ch#',ch);
  text(text_x,text_y,ch,'units','normalized','color',color_order(i))
  text_y=text_y-0.05;
  hold on;
```

```
end
ylabel('measurement');
grid;
title('Redundant Sensors');
varPlot=varPlot+1;

%plotWindowVar.m

x_size=size(XX);
figure(varPlot);
text_x=0.9;
text_y=0.9;
for i=1:x_size(2)
   plot(XX(:,i),color_order(i));
   ch=num2str(i);
   ch=strcat('ch#',ch);
   text(text_x,text_y,ch,'units','normalized','color',color_order(i))
   text_y=text_y-0.05;
   hold on;
end
ylabel('measurement');
grid;
title('Redundant Sensors');
varPlot=varPlot+1;
```

A.2.2 Moving Window Function

```
%r2mw.m
%Script to calculate moving window ICA
%Developed by Jun Ding
%2003

if ~isempty(inputX)
   diChoice = 2;
   bd = 16000;
   sig = 2;
   param = 2;
   ed = 37913;
   sr = 15;
   drift_method=diChoice;
   perc_method=1;
   ref=0;
   if diChoice==7
      ccv=param;
   elseif diChoice==10;
```

```
      sigma=param;
   else
      perc=param;
      ccv=1;
      sigma=1;
   end


driftX=simulate_drift(inputX,sig,perc,perc_method,ref,bd,ed,sr,drift_method,ccv,sigma);
   XX=driftX;
end

xx1=driftX;
dim=size(xx1);
ica_coef=zeros(8,8,dim(2));
data_std=zeros(8,8,dim(2));
data_corr=zeros(8,8,dim(2));
for j = 1: 8
   for i = 1:8
      if i>=j
         a=i*4000;
         b=(j-1)*4000+1;
         xx=xx1(b:a,:);
         ss=std(xx);
         icNum=min(4,dim(2));

[Y,A,W]=FASTICA(xx','approach','symm','g','tanh','displayMode','off','verbose','off','num
OfIC',icNum);
         if ~ isempty(Y)
            [W1, s_IC,xavg,index] = icapost(xx',Y,W);
            cw(j,i)=cond(W1);
            m_cor=corrcoef([s_IC' xx]);
            for k=1:dim(2)
               ica_coef(j,i,k)=W1(index,k);
               data_std(j,i,k)=ss(k);
               data_corr(j,i,k)=m_cor(1,k+1);
            end
            ica_std(j,i)=std(s_IC);
         end
      end
   end
end

dd=ones(1,dim(2));
data_sum_err=zeros(8,8);
```

```
for i=1:8
   for j=1:8
      if ~((data_std(i,j,1)==0))
        c2(1,:)=data_std(i,j,:);
        d2(1,:)=data_corr(i,j,:);
        p=polyfit(c2,d2,1);
        y=polyval(p,c2);
        ds=(d2-y).^2;
        data_sum_err(i,j)=ds*dd';
      end
   end
end
```

A.2.3 ICMP Calculation

```
%ICMP Script
% Jun Ding
%2003
load VCPCRMODEL
X=[yp ytest1 ytest2];
[estimate,ACvalues,estimate_weights,est_dev,acceptance,consistency]=icmp3(X);
figure(1)
plot(X)
hold on
plot(estimate,'k-')
legend('Infer Sensor','ch#1','ch#2','ICMP est')
title('ICMP results improved with inferential sensor')

res1=estimate-ytest1;
res1=res1-mean(res1(1:100,1))*ones(size(res1,1),1);
figure(2)
plot(res1);
axis([1 7000 -15 15])
title('residual of ch#1,ICMP+Inferential Sensor')

res2=estimate-ytest2;
res2=res2-mean(res2(1:100,1))*ones(size(res2,1),1);
figure(3)
plot(res2);
axis([1 7000 -15 15])
title('residual of ch#2,ICMP+Inferential Sensor')
```

```
function
[estimate,ACvalues,estimate_weights,est_dev,acceptance,consistency]=icmp3(X);

% ICMP algorithm for redundant channel calibration verification


% INPUTS
% X is the redundant data
%   Data should be entered as (samples x signals)
% IMPORTANT:  In this function, the consistency and acceptance criteria are estimated
based on the
%   deviation of the signals from their mean values.  This is not a fail-safe method of
estimating these
%   parameters, I just found it works most of the time.  The EPRI documents do not
provide any automatic
%   determination of these parameters, leaving it up to the user.  Better results may be
obtainable by adjusting
%   these parameters.
% OUTPUTS
% estimate is the parameter estimate of the ICMP algorithm
% ACvalues are the alarm values, 0 or 1, for each observation for all channels, thus its
dimensions are the same as
%   the data X.
% estimate_weights are the weights for each channel for each observation that are used to
compute the estimate
% est_dev is a matrix of differences between the estimate and each signal contained in X.
% acceptance returns the acceptance criteria value for each channel. These are the values
that are compared to the absolute deviation
%   between the estimate and each channel to determine out-of-calibration conditions.
% consistency returns the consistency values for each channel. These are the values that
are compared to the deviations between channels
%   to determine the estimate_weights.




[np,ns]=size(X);
data=X;

deviation=data-((mean(data')')*ones(1,ns));
avgdeviation=abs(mean(deviation));
stddeviation=std(deviation);
consistency=avgdeviation+3*stddeviation;
acceptance=avgdeviation+4*stddeviation;
```

```
if (ns==(length(consistency))&ns==(length(acceptance)))
else
   error('The number of columns (signals) in data must be equal to the length of the
consistency and acceptance vectors')
end;

for k=1:ns;
   %divide data into signal k, and remainder
   d1=data;
   d2=d1(:,k);
   d1(:,k)=[];
   %compute absolute deviation between signal k and remaining signals
   absdiff=abs(d1- d2*ones(1,(ns-1)));
   %divide consistency values for signal k and remainder
   c1=consistency;
   c2=c1(k);
   c1(k)=[];
   %compute consistency sums for all absolute deviations
   c3=c1+(c2*ones(1,(ns-1)));
   %determine consistency for each absolute deviation
   cc=absdiff<(ones(np,1)*c3);
   %sum to determine the consistency of signal k for each sample
   C(:,k)=(sum(cc'))';


end;

   %Compute total consistency for each sample
   Ctot=sum(C')';
   %weight each sample by the consistency values determined
   est=(data.*C);
   %Normalize the estimate by the total consistency
   estimate=(sum(est')')./Ctot;

   estimate_weights=C./ (Ctot*ones(1,ns));


   % Estimate Deviations

   abs_est_dev=abs(data-(estimate*ones(1,ns)));

   % Create an npXns alarm matrix
```

```
ACvalues=abs_est_dev> (ones(np,1)*acceptance);

est_dev=(data-(estimate*ones(1,ns)));
```

A.2.4 PCR Calculation

```
% PCR Script
% Developed by Jun Ding
% 2004
load T0308
load T0401
dim=size(T0401);
x=T0308;
xt=T0401;
x=[x ones(37441,1)];
xtrain=x(1:10000,4:31);
ytr1=x(1:10000,2);
ytr2=x(1:10000,3);
xtest=xt(1:dim(1),4:31);
ytest1=xt(1:dim(1),2);
ytest2=xt(1:dim(1),3);

options.display='off';
options.plots='none';
model1=pcr(xtrain,ytr1,1,options);
pred1=pcr(xtest,model1,options);
yp1=pred1.pred;
yp_m1=yp1{1,2}(:,1);
figure(10)
plot(yp_m1)
hold on
plot(ytest1,'r-')
legend('prediction','ch#1');
title('PCR prediction for turbine first stage pressure 2003-08 - 2004-03')
res1=yp_m1-ytest1;
res1=res1-mean(res1)*ones(size(res1,1),1);
figure(11)
plot(res1);
axis([1 dim(1) -10 10])
title('residual of ch#1,PCR')

model2=pcr(xtrain,ytr2,1);
pred2=pcr(xtest,model2);
yp2=pred2.pred;
yp_m2=yp2{1,2}(:,1);
```

```
figure(12)
plot(yp_m2)
hold on
plot(ytest2,'r-')
legend('prediction','ch#2');
title('PCR prediction for turbine first stage pressure 2003-08 - 2004-03')
res2=yp_m2-ytest2;
res2=res2-mean(res2)*ones(size(res2,1),1);
figure(13)
plot(res2);
axis([1 dim(1) -10 10])
title('residual of ch#2,PCR')
```

A.2.5 Bootstrap Prediction Error Interval for Linear Model

```
%Linear Model Bootstrap Prediction Error Interval Script
%Developed by Jun Ding
%2004
x=randn(85,1);
y=2*x+0.9*randn(85,1);
b=regress(y,x)
y_hat=b*x;
figure(1)
plot(x,y_hat,x,y,'r+')
legend('yhat','y')
xlabel('x')
ylabel('y')
dim=size(x);
m=100;
for i=1:m
   rnd_index=floor(dim(1).*rand(1,dim(1)))+1;
   for j=1:dim(1)
      xx(j,1)=x(rnd_index(1,j),:);
   end
   y=2*xx+0.9*randn(85,1);
   b=regress(y,xx)
   y_hat=b*xx;
   res(:,i)=(y_hat-y);
   i=i
end
m=mean(res');
st=std(res');
figure(2)
errorbar(m,2*st);
```

```
xlabel('data points')
ylabel('residual')
```

A.2.6 ICA Bootstrap Prediction Error Interval

```
% Bootstrap Interval for ICA
% Developed by Jun Ding
% Jun Ding

load T0401_ICA
dim=size(z);
icNum=min(4,dim(2));
m=1000;
ica_coef=zeros(m,dim(2));
xx=z;
for i=1:m
   rnd_index=floor(dim(1).*rand(1,dim(1)))+1;
   for j=1:dim(1)
      xx(j,:)=z(rnd_index(1,j),:);
   end

[Y,A,W]=FASTICA(xx','approach','symm','g','tanh','displayMode','off','verbose','off','num
OfIC',icNum);
   if ~ isempty(Y)
       [W1, s_IC,xavg,index] = icapost(xx',Y,W);
        for k=1:dim(2)
           ica_coef(i,k)=W1(index,k);
        end
     end
     i=i
end
res=zeros(2,3,dim(1));
for i=1:dim(1)
   for j=1:m
     est=z(i,:)*ica_coef(j,:)';
     r1(j)=est-z(i,1);
     r2(j)=est-z(i,2);
     r3(j)=est-z(i,3);
   end
   res(1,1,i)=mean(r1);
   res(2,1,i)=std(r1);
   res(1,2,i)=mean(r2);
   std2=std(r2);
   res(2,2,i)=std2;
   p2(i)=2*std2/z(i,2);
```

```
    res(1,3,i)=mean(r3);
    std3=std(r3);
    res(2,3,i)=std3;
    p3(i)=2*std3/z(i,3);
    i=i
end

errorbar(res(1,2,:),res(2,2,:)*2)
legend('ch#2')
title('Bootstrape 95% confidence interval')
ylabel('residuals')
xlabel('time')

hist(r2,100)
xlabel('residual')
ylabel('frequency')
title('Bootstrap residual of ch#2 at point 44641')
```

Vita

Jun Ding, originally from Shanghai, China, earned a Bachelor of Science degree in Nuclear Physics and Technology from University of Science and Technology of China in 1992. He joined Shanghai Nuclear Engineering Research and Design Institute, China in July, 1992. He was an instrumentation and control engineer and has designed NPP radiation monitoring system along with various other engineering projects. The NPP whole range simulator project was awarded the first prize of science and technology achievement of DOE of China in 2000. He has finished graduate course in Electrical Engineering from Shanghai Jiaotong University in 1999. He then received a M.S in nuclear engineering from the University of Cincinnati in 2000 and a M.S in Physics Entrepreneurship Program from Case Western Reserve University in 2002. He entered the University of Tennessee to work toward a Doctor of Philosophy degree in Nuclear Engineering in the fall of 2002. While at the university, he conducted research on redundant sensor validation using independent component analysis. He received his Ph. D in Nuclear Engineering in August, 2004.