LINEAR AND NONLINEAR INTERVAL MULTISCALE PCA BASED FAULT DETECTION METHODS

A Thesis

by

MOHAMED NASIR OMER

Submitted to the Office of Graduate and Professional Studies Of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee, Mohamed Nounou
Co-Chair of Committee, Hazem Nounou
Committee Member, Ahmed Abdel-Wahab

Head of Department, Arul Jayaraman

August 2020

Major Subject: Chemical Engineering

Copyright 2020

ABSTRACT

Statistical process control is an integral set of techniques in chemical engineering that can help guarantee the effective operation of various engineering systems. A multitude of techniques have been and are still being developed in order to detect and monitor faults in processes using process data. Deviations from the normal operating conditions can cause a myriad of abnormalities in the data, such as temperature and pressure readings, which can then be used to detect faults in the process. Processes variables can be linearly or non-linearly correlated, and the aim of this work is to start off with linearly correlated data, and then move on to non-linearly correlated data, which is more complex and requires more advanced methods to deal with. The basis of fault detection when the data is linearly correlated is principal component analysis, which is widely used. Two extensions of this method, interval PCA (IPCA) and multiscale PCA (MSPCA), have been developed to help increase the efficiency of fault detection. In MSPCA, the data are decomposed using wavelets at multiple scales, and PCA is applied at each scale before reconstructing the data back to the time domain, where PCA is applied again to detect faults. MSPCA helps deal with auto-correlated noise and reduces its effect on the accuracy of fault detection. Interval PCA, on the other hand, is a technique that helps deal with uncertainty in the data by converting the single valued data interval by aggregating the measured samples over a time horizon, and PCA is applied on the generated intervals. These two methods will be combined, and new fault detection method, called interval multiscale PCA (IMSPCA), will use the advantages of both to have more efficient fault detection method. For the nonlinear case, a neural network-based modification of the algorithm will be used to

developed neural network IMSPCA (NNIMSPCA). Neural networks are a group of techniques, modeled after neurons in the human brain that are taught to recognize complex patterns in data.

ACKNOWLEDGMENTS

I would like to thank Dr. Mohamed Nounou for his support and express my gratitude towards him. I would also like to acknowledge Dr. Hazem Nounou, my co-chair, and Dr. Ahmed Abdel-wahab, committee member.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

The committee that supervised this work consisted of Dr. Mohamed Nounou, from the Chemical Engineering Department, chair of the committee, Dr. Hazem Nounou, from the Electrical Engineering Department, co-chair of the committee, and Dr. Ahmed Abdel-Wahab, from the Chemical Engineering Department, committee member.

Funding Sources

The first year of this work was funded by a graduate assistantship from the Graduate Studies Office at Texas A & M at Qatar, and the second year was funded by an Itochu Fellowship.

NOMENCLATURE

CPV	Cumulative Percent Variance
FAR	False Alarm Rate
FDR	Fault Detection Rate
IMSPCA	Interval-Multiscale Principal Component Analysis
IPCA	Interval Principal Component Analysis
m	Number of Data Samples
MSPCA	Multiscale Principal Component Analysis
n	Number of Variables
NNIMSPCA	Neural Network Interval-Multiscale Principal Component Analysis
PCA	Principal Component Analysis
SNR	Signal-to-Noise Ratio
SPC	Statistical Process Control

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

1.1 Overview

In the chemical engineering industry, the field of statistical process controls (SPC) is an ever growing an expanding field, which deals with diagnosing and detecting faults. It is estimated that an average of $20 billion is lost every year in just the petrochemical industry in the United States due to faulty SPC [1]. Statistical process control has been constantly developing since the 1920's, with the advent of the first control chart developed by Walter Shewhart [2].The main aim of SPC is to monitor the process and ensure that it remains within the defined limits. The Shewhart chart utilizes data from the most recent samples and calculates a threshold, usually $m \pm 3\sigma$, where m is the mean and σ is the standard deviation of the data, and determines what could be considered to be a fault and needs further investigation. This very simplistic approach was the basis of control charts; however, it was very quickly proven to be inefficient in detecting small shifts in processes and correlation between variables. The focus of this work is on fault detection, which is a branch of SPC. Deviations in the operating conditions or equipment in a chemical plant may cause abnormalities in the data, and it is these abnormalities that need to be detected in order for a fault to be declared [3].

In a normal operating process, up to 1500 variables maybe observed at a time [1]. Handling "Big Data" has been a topic of interest in machine learning and data science [4], and the use of computational machine learing tools in data processing will be used in this work. Fault detection methods that deal with such processes are divided into 3 categories: quantitative model-based methods, qualitative model-based, and process data model-

based methods. Quantitative based methods require a deep understanding of the process, while qualitative based methods rely on rules set by experts in the process being monitored [5]. These requirements make them difficult to use with highly complex data. Process data-based methods, however, only use past data from the process in order to develop the models needed for fault detection. Principal Component Analysis (PCA) is one of such process data-based methods [6]. It is a linear dimensionality reduction tool that has been used in process monitoring for a long time, where data that is linearly correlated can be decorrelated and its dimensionality can be reduced. PCA is also used to filter out noise and discover features and dynamics in data [7]. The goal of PCA is to reconstruct the data using less dimensions than the original data, while retaining most of the variability in the data. In order to perform PCA, the covariance matrix of the data is calculated, and the eigenvectors and eigenvalues of this matrix are then determined. The eigenvalue of corresponding to each variable gives information for how much variance is explained by the variable. A suitable number of principal components (PC's) is then chosen, and the data is reconstructed using the eigenvectors of the corresponding PC's. The Q statistic is usually then calculated, and a threshold is drawn that is used to determine where the faults are [8].

New algorithms for fault detection are constantly being developed in order to minimize safety hazards and costs, and maximize efficiency of operation in chemical plants. This work aims to combine two existing methods, interval PCA (IPCA) and multiscale PCA (MSPCA), in order to develop a better method that can be used on linear datasets, interval multiscale PCA (IMSPCA) . IMSPCA will then be modified using neural

networks, to develop an algorithm that can be used for nonlinear datasets, neural network IMSPCA (NNIMSPCA).

2. IMPROVING FAULT DETECTION USING IMSPCA

2.1 PCA

2.1.1 Theory

Principal component analysis (PCA) is a multivariate statistical technique that is used on data to reduce its dimensionality and extract important information from it. It was developed by Pearson in 1901 as a way to represent data with a new axis that better fit the system of points [9]. PCA transforms the original data using a new set of basis called principal components (PC) which capture the maximum variation and minimize redundancy and noise. The benefits of PCA is that while most of the variability in the data is retained, there is a minimal loss in information if the correct number of PC are chosen. At the same time, the obtained PC are not correlated with each other and can be studied individually without affecting the rest [10].

PCA represents a normalized data matrix X, with *m* samples and *n* variables, as the product of two matrices, $T \in R^{n \times p}$ and $P \in R^{m \times p}$, which contain the PC and loading vectors of X respectively, as $X = TP^T$. The number of principal components and corresponding eigenvectors that are retained are represented by rewriting TP^T as [11],

$$X = \hat{T}\hat{P}^T + \tilde{T}\tilde{P}^T, \quad (1)$$

where the $\hat{T}$ and $\hat{P}$ matrices contain the retained principal components and eigenvectors, respectively, and $\tilde{T}$ and $\tilde{P}$ contain the remaining principal components and eigenvectors, respectively. The main goals of PCA are to obtain the most important features of the data,

compress the amount of data while retaining the important information, and to examine the structure of the variables and check for correlations, and this is done by choosing a number of PC that is less than the number of variables in the original data set [12].

Determining the number of important PC is a critical step in PCA modeling, as underestimating the number of PC leads to an incomplete representation of the data, and overestimating leads to an over defined and noisy reconstruction. Several techniques have been developed for this purpose, which include the scree plot, the variance of the reconstruction error, and cumulative percent variance (CPV).

- The scree plot graphs eigenvalues of the correlation matrix of the data in decreasing order versus the variable index, and a suitable number is chosen at the point where the graph becomes a straight line [13].

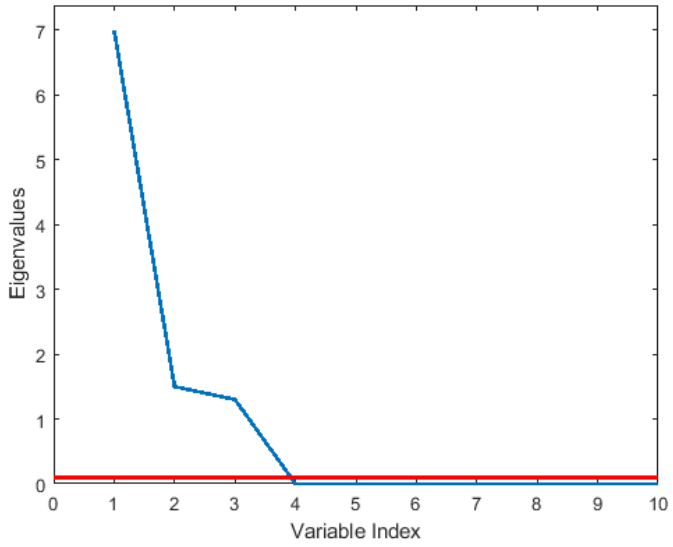

Figure 1: Scree plot of 10 correlated variables

- Using the variance of reconstruction error method, the VRE is plotted for each possible number of PC, and the point at which the VRE is at minimum is chosen to correspond to the optimal number of PC [14].

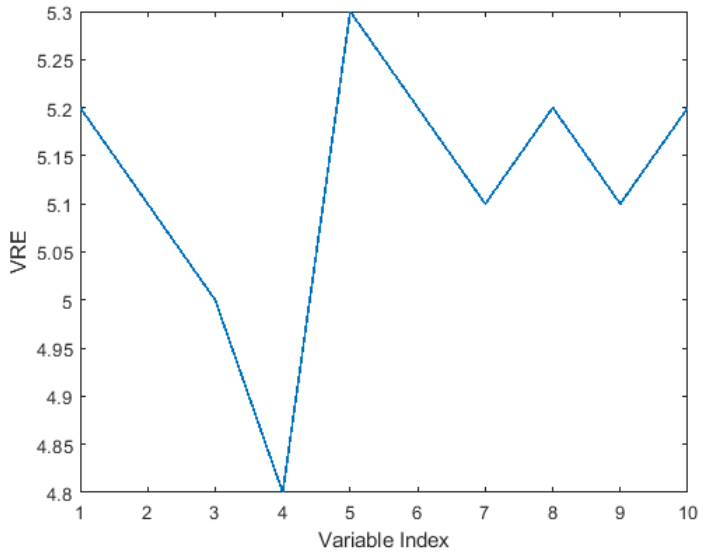

Figure 2: VRE for 10 variables with the minimum being at 4 PC

- Using CPV, a desired percentage of variability captured by the PC (e.g. 90% or 95%) is chosen, and the number of PC corresponding to that percentage is then retained. [15]

$$CPV(l) = \frac{\sum_{i=1}^{l} \lambda_i}{trace(\sum)} \text{x } 100$$

This is the most commonly used method to calculate the retained number of PC's and will be used in this work. [16]

2.1.2 Applications of PCA

PCA is a widespread technique used to reconstruct data with fewer components while retaining most of the variability and filtering out noise. It has been used in the field of material science to uncover patterns in atoms and molecules and then determine the correlation of such patterns with material characteristics [17]. It is also used in the field of chemometrics, a subfield of biochemistry that deals with applying multivariate statistical methods to laboratory studies [18], and in electrocardiography (ECG) signal processing to reduce the size of the signals so that they can be compressed [19]. In chemical engineering, the vast amounts of data that are generated from plants are compressed using PCA, which can then be used to determine correlations and calculate fault detection metrics for fault detection.

2.1.3 PCA algorithm

1. The training data set is first converted to an *m x n* data matrix X, where *m* is the number of data samples and *n* is the number of variables. In order to keep the data internally consistent, the data is first standardized. This ensures that each variable has the same mean of zero and standard deviation of one, so that they can be easily compared. This is useful when the data set contains variables of different natures, i.e. temperature and pressure readings [20]. To standardize the data, the mean, μ, and standard deviation, σ, of each variable in the training set is calculated. Then, the mean is subtracted from each data point, and divided by the standard deviation

of its data set. This ensures that that the mean of each variable is zero, and the standard deviation is one.

2. Calculating the correlation matrix of the data. The correlation matrix is a square matrix where the diagonals are calculated to be the variances of each variable, and the other components are the covariances, and is useful in assessing the correlation between each two variables in a data set [20]. In the m x n data matrix X, the covariance matrix, V can be given by

$$V = \begin{pmatrix} \sigma(x_1, x_1) & \sigma(x_2, x_1) & \cdots & \sigma(x_{n-1}, x_1) & \sigma(x_n, x_1) \\ \sigma(x_1, x_2) & \sigma(x_2, x_2) & \dots & \sigma(x_{n-1}, x_2) & \sigma(x_n, x_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma(x_1, x_{n-1}) & \sigma(x_2, x_{n-1}) & \dots & \sigma(x_{n-1}, x_{n-1}) & \sigma(x_n, x_{n-1}) \\ \sigma(x_1, x_n) & \sigma(x_1, x_n) & \cdots & \sigma(x_{n-1}, x_n) & \sigma(x_n, x_n) \end{pmatrix} \quad (2)$$

In the case of standardized data, the diagonals of the correlation matrix will be one, giving the covariance matrix.

3. Finding the eigenvalues and eigenvectors of V. The eigenvalues are important because they explain the percent of variance explained by each variable and are used to assess how many eigenvectors to be retained. The eigenvalues, Λ, are calculated and arranged in a diagonal matrix,

$$\Lambda = \begin{pmatrix} \Lambda_1 & 0 & \cdots & 0 & 0 \\ 0 & \Lambda_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \Lambda_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & \Lambda_n \end{pmatrix} \quad (3)$$

The eigenvalues are arranged in decreasing order, and using either of the methods of Section 2.1 a suitable number is chosen. The corresponding eigenvectors are

then used to transform the data into the principal component space. This is shown in Figure 3, where *c* eigenvectors are chosen.

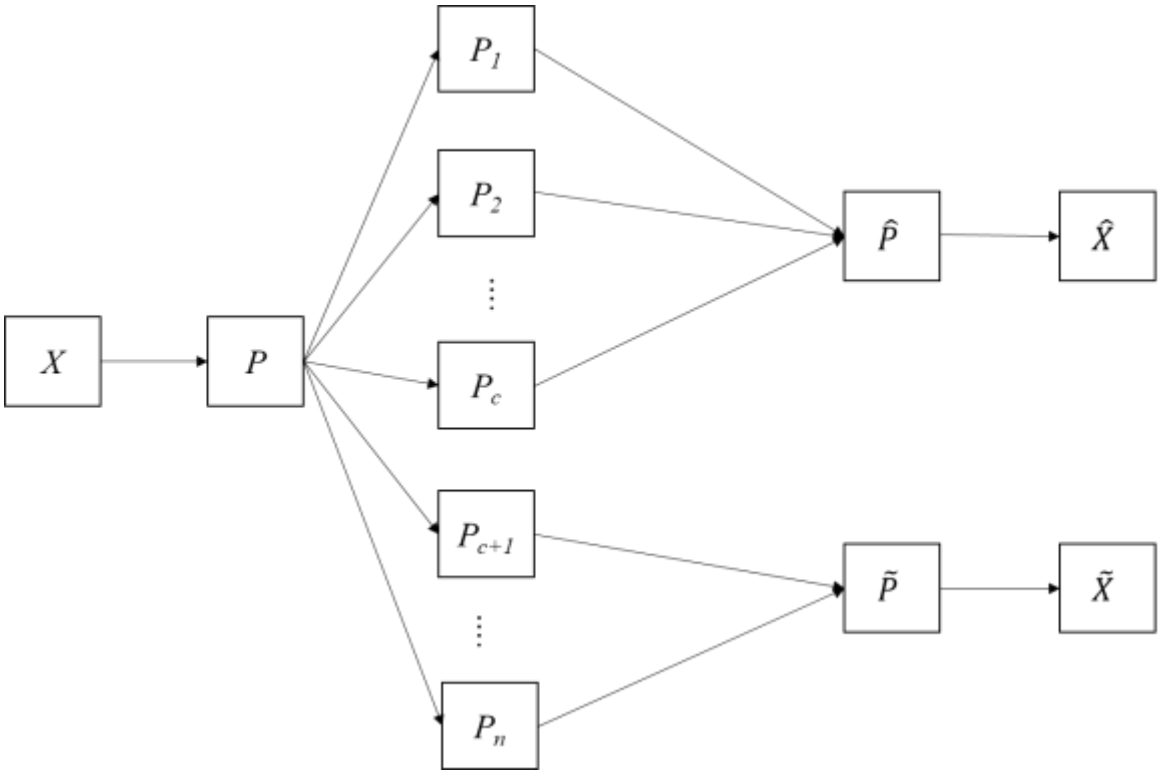

Figure 3: Figure showing retention of principal components

4. The transformation matrix $\hat{C} = \hat{P}\hat{P}^T$ is used to find the predicted data matrix $\hat{X} = X\hat{C}$ and residuals matrix $\tilde{X} = X(I - \hat{C}) = X\tilde{C}$.
5. $\hat{C}$ and $\tilde{C}$ are used to calculate the predicted and residual matrices of the testing data, Y, using $\hat{Y} = Y\hat{C}$ and $\tilde{Y} = Y\tilde{C}$, respectively.

2.1.4 Fault Detection using PCA

When a fault occurs in one of the variables, the PCA model will contain differences in $\hat{Y}$ and $\tilde{Y}$ that are larger than in normal operations. The two most commonly used statistics in quantifying these residuals are the Q-statistic and the Hotelling's T^2-statistic. The T^2 statistic is a measure of the variation of each sample in the retained PC subspace, and is given by the equation,

$$T_i^2 = x_i P \Lambda^{-1} P^T x_i^T \tag{4}$$

The Q-statistic (or squared prediction error) is a calculation of the difference between a measurement and its corresponding projection in the PCA subspace[21]. The equation for the Q-statistic for a measurement x_i is,

$$Q_i = \tilde{x}_i \tilde{x}_i^T = x_i (I - PP^T) x_i^T = x_i \tilde{C} x_i^T \tag{5}$$

where $\tilde{x}_i$ represents the projection of the measurement. The Q-statistic will be used in this work as it is more sensitive than the T^2 statistic, which requires a larger shift in the process to detect a fault [22].

After calculating the T^2 or Q statistics, the accuracy of fault detection is evaluated using two main criteria: fault detection rate (FDR) and false alarm rate (FAR). FDR is the percentage of successfully detecting faults when faults exit, while FAR is the percentage of detecting faults when they do not exist [9]. After calculating the Q statistic, a threshold is then calculated.

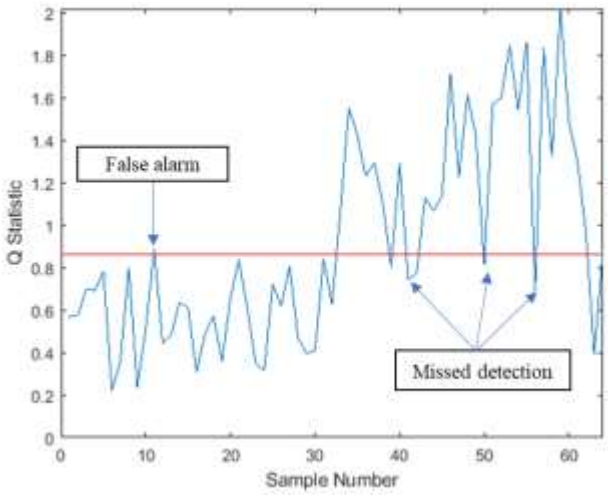

Figure 4: Figure showing false alarm and detection rate

2.2 Multiscale Principal Component Analysis (MSPCA)

Multiscale principal component analysis (MSPCA) is one of the extensions of PCA that uses a multiscale representation to represent the data by passing it through low and high pass filters derived from scaling and basis wavelet functions, respectively. This is done to extract important features from the signal and remove noise, with the purpose of improving fault detection. Data can be passed through linear filters such as Shewhart, exponentially weighted moving average (EWMA), and cumulative average (CUSUM)[23], but their disadvantage lies in the fact that they can only look at data at one frequency, and only have good performance when the variables are gaussian and not correlated. MSPCA uses nonlinear filters derived from wavelet and scaling functions, which can separate the deterministic and stochastic features of a signal, and which have been shown to improve fault detection for multivariate data.

2.2.1 Multiscale representation of data

In multiscale decomposition, the signal is written as the sum orthonormal basis functions called wavelets and scaling functions, that are localized in both time and frequency[24]. There are various families of wavelet functions such as Haar, Daubechies, and Coiflet. In this work, the Haar wavelet function is used for its simplicity and ease of use [25].It is also the only family of wavelets that is symmetric and at the same time compactly supported and orthogonal [26]. The signal is convoluted with a low pass filter derived from a scaling function, which has the form,

$$\varphi_{jk}(t) = 2^{\frac{-j}{2}}\varphi(2^{-j}t - k) \tag{6}$$

and shape,

Figure 5: Shape of the Wavelet Scaling Function Used in multiscale decomposition

where the parameters j and k represent the dilation and translation parameters, respectively. The first detail signal is the difference between the original signal and the approximate signal, and is obtained by convoluting the original signal with a high pass filter that is derived from a wavelet function which has the form,

$$\psi_{jk}(t) = 2^{\frac{-j}{2}}\psi(2^{-j}t - k) \tag{7}$$

and shape,

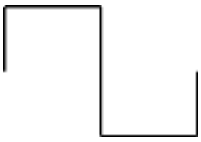

Figure 6: Shape of the Wavelet Basis Function used in multiscale decomposition

This process is repeated on every scale by applying the filters on the previous approximate signal, until a certain depth J is reached. The original signal can be expressed mathematically as the sum of the last approximate signal, and all the detailed signals as,

$$x(t) = \sum_{k=1}^{n2^{-j}} a_{jk}\varphi_{jk}(t) + \sum_{j=1}^{j}\sum_{k=1}^{n2^{-j}} d_{jk}\psi_{jk}(t). \quad (8)$$

where n is the number of samples.

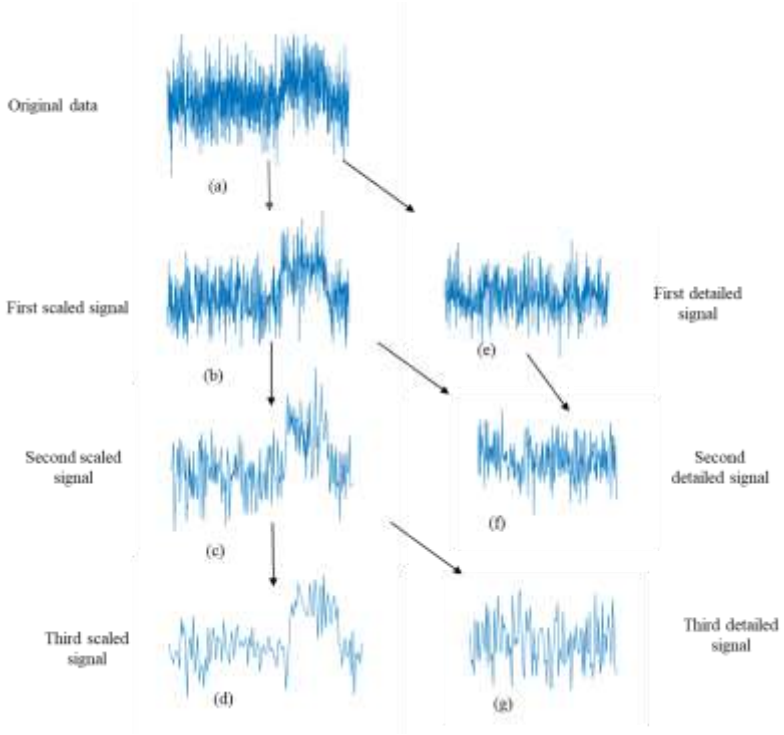

Figure 7: Signal decomposed to a depth of 3

Multiscale decomposition requires defining a decomposition depth, J. A very large depth may result in important information being eliminated, while a small depth may result in the retention of large amount of noise. The optimum rule set for determining depth for a signal of length *m* is a heuristic presented by Bakshi, using the equation $J = log_2 m - 5$, which results in a depth where the coarsest scale has 32 coefficients[24].

2.2.2 MSPCA algorithm

The wavelet decomposition shown in section 2.2.1 is combined with PCA to obtain MSPCA, and the algorithm is shown in Figure (6).

1. After standardizing the training data, X, and the testing data, Y, compute the wavelet decomposition of each signal to a depth, J.
2. Collect and group the approximate signals and each detail signal at the different depths from each variable, and compute the covariance matrix for each set.
3. Apply PCA, and then use the model to compute the Q statistic of the training and testing data sets.
4. For the training data, if any of the Q statistic values cross the threshold, all of the scale is retained.
5. For the testing data, if any of the Q statistic values do not cross the threshold, the sample is zeroed on the original scale
6. Reconstruct the data back into the time domain after applying the above rules, and apply PCA.
7. Calculate the Q statistic and assess the fault detection rate and false alarm rate.

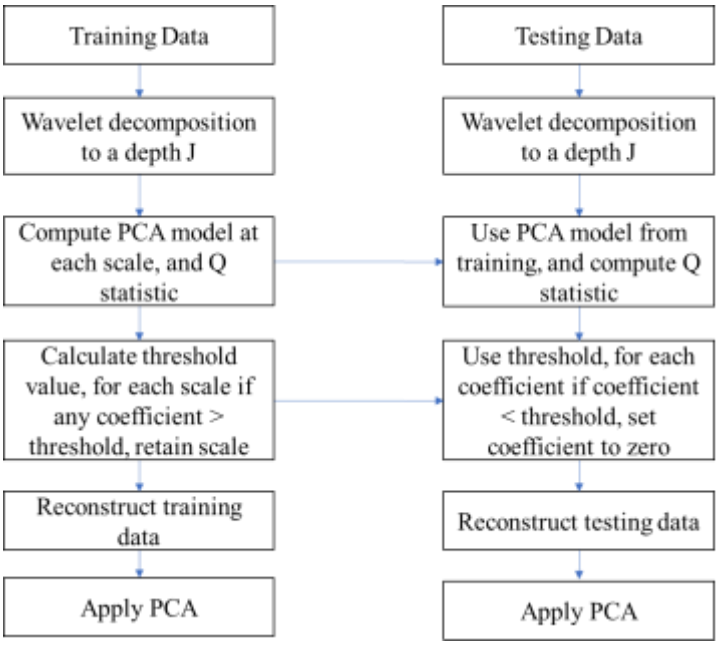

Figure 8: Schematic for MSPCA

2.3 Interval principal component analysis (IPCA)

Interval PCA is an extension of PCA that is applied to data sets with a high number of samples and/or uncertainties in the data, i.e. gaps in some readings [27]. The basis of this extension is the aggregation of data into useful groups that make it more computationally efficient to work with.

2.3.1 Interval Generation

An interval is constructed using a lower and upper bound, for example [a,b], such that $a \leq$ b and { a,b } $\in \mathbb{R}$. A visual representation of an interval dataset is shown below [28].

$$|\boldsymbol{X}| = \begin{bmatrix} [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{1,1}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{1,1}}] & [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{1,2}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{1,2}}] & \dots & [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{1,p}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{1,p}}] \\ [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{2,1}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{1,1}}] & [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{2,2}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{2,2}}] & \dots & [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{2,p}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{2,p}}] \\ . & . & \dots & . \\ . & . & \dots & . \\ [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{n,1}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{n,1}}] & [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{n,2}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{n,2}}] & \dots & [\boldsymbol{x}^{\boldsymbol{l}}_{\boldsymbol{n,p}}, \boldsymbol{x}^{\boldsymbol{u}}_{\boldsymbol{n,p}}] \end{bmatrix} \tag{9}$$

where $x^{l}_{n,p}$ represents the lower bound of a given interval, and $x^{u}_{n.p}$ represents the upper bound of the same interval. Then for each interval, the center, x_c is calculated as

$$x_C = \frac{x_l + x_u}{2} \tag{10}$$

and the radius, x_r is calculated as

$$x_r = \frac{x_u - x_l}{2} \tag{11}$$

In some cases, the center is taken as the average of the 2 extremes, while the radius is taken as the standard deviation of the 2 extremes in the interval. After this is done, two new data sets are generated: one containing all the centers, and one containing all the radii. They are represented by $\{X^C, X^R\}$, which are shown below.

$$[X^C] = \begin{bmatrix} \left[\frac{x^l_{1,1} + x^u_{1,1}}{2}\right] & \left[\frac{x^l_{1,2} + x^u_{1,2}}{2}\right] & \dots & \left[\frac{x^l_{1,m} + x^u_{1,m}}{2}\right] \\ \left[\frac{x^l_{2,1} + x^u_{1,1}}{2}\right] & \left[\frac{x^l_{2,2} + x^u_{2,2}}{2}\right] & \dots & \left[\frac{x^l_{2,m} + x^u_{2,m}}{2}\right] \\ . & . & \dots & . \\ . & . & \dots & . \\ \left[\frac{x^l_{n,1} + x^u_{n,1}}{2}\right] & \left[\frac{x^l_{n,2} + x^u_{n,2}}{2}\right] & \dots & \left[\frac{x^l_{n,m} + x^u_{n,m}}{2}\right] \end{bmatrix} \tag{12}$$

$$[X^R] = \begin{bmatrix} \left[\frac{x_{1,1}^u - x_{1,1}^l}{2}\right] & \left[\frac{x_{1,2}^u - x_{1,2}^l}{2}\right] & \cdots & \left[\frac{x_{1,m}^u - x_{1,m}^l}{2}\right] \\ \left[\frac{x_{2,1}^u - x_{1,1}^l}{2}\right] & \left[\frac{x_{2,2}^u - x_{2,2}^l}{2}\right] & \cdots & \left[\frac{x_{2,m}^u - x_{2,m}^l}{2}\right] \\ \vdots & \vdots & \cdots & \vdots \\ \left[\frac{x_{n,1}^u - x_{n,1}^l}{2}\right] & \left[\frac{x_{n,2}^u - x_{n,2}^l}{2}\right] & \cdots & \left[\frac{x_{n,m}^u - x_{n,m}^l}{2}\right] \end{bmatrix} \quad (13)$$

There are three types of interval PCA: Centers IPCA (CIPCA) [29], Midpoint-radii IPCA (MRIPCA) [30], and symbolic covariance IPCA (SIPCA) [31]. The method chosen in this work is MRIPCA, was shown to be better at detecting faults than the other two methods[28].

2.3.2 Midpoint-Radii IPCA (MRIPCA)

In MRIPCA, PCA is applied independently on the centers and radii data, and each corresponding PCA model is used for fault detection. The choice of how many samples to aggregate per interval are dependent on two things: the level of uncertainty in the data, and the sample size of the data.

1. The training data sets X^C and X^R are used to compute the transformation model matrices $\{\hat{C}^C, \tilde{C}^C\}$ and $\{\hat{C}^R, \tilde{C}^R\}$, by using $\hat{C} = \hat{P}\hat{P}^T$ and $\tilde{C} = (I - \hat{P}\hat{P}^T)$, which can be used to calculate the Q statistics.
2. $\{\hat{C}, \tilde{C}\}$ are used on the testing data $\{Y^C, Y^R\}$ to generate their transformation matrices to calculate the Q statistic: $\{\tilde{Y}^C, \tilde{Y}^R\} = \{Y^C\tilde{C}^C, Y^R\tilde{C}^R\}$.
3. A false alarm rate is then set and that threshold value is used on the faulty testing data to assess fault detection rate.

2.4 Objectives and methodology

In this section the parameters used to assess the different algorithms are discussed. The objectives of this work as follows:

1. Develop a combined interval-multiscale PCA (IMSPCA) algorithm that can detect faults in linear process data with a high accuracy.
2. Perform a comparative study of IMSPCA with the conventional PCA, IPCA, and MSPCA algorithms.

2.4.1 Interval-Multiscale principal component analysis

In literature, there is no observed integration of wavelet decomposition with interval data, which is why this topic was explored. The benefits of generating intervals such as reduced computational complexity and reducing the data set size, coupled with the multiscale decomposition, which can separate the stochastic and deterministic features of a signal, is carried out in this work to yield Interval-Multiscale principal component analysis (IMSCPA). Using IMSPCA, the data is first converted into centers and radii intervals, and the multiscale decomposition to a certain depth J is applied on the interval data. The algorithm is shown in section 2.4.2.

2.4.2 Algorithm

1. The centers and radii interval matrices $\{X^C, X^R\}$, are generated from the data matrix X. The rest of the steps are done independently on the centers and radii data.

2. Multiscale decomposition is performed using the scaling and basis functions discussed in section 2.2.2.
3. PCA is done each group of scaled and approximate signals, and the Q statistic is calculated.
4. The reconstruction rules discussed in section 2.2.2 are applied, and then the data is reconstructed back into intervals.
5. IPCA is performed on the reconstructed data sets.
6. The Q statistic is then calculated, and a threshold is set to assess the FDR and FAR.

2.5 Data set generation

The data sets that are simulated in this work are of two types: one where the dependent variables are linearly correlated to the independent variables (cases 1 and 2), and one where the dependent variables are nonlinearly correlated to the dependent variables (cases 3 and 4). This section will cover cases 1 and 2, and then IMSPCA will modified and tested in section 3. In each case, the first two variables generated are random Gaussian variables with a certain mean, μ_i, and standard deviation, σ_i. The data is divided into two: training and testing. Each of these sets contain 32,768 samples, and this is purposefully done in order to have large data sets where the advantages of interval generation can be felt.

2.5.1 Fault generation

In processes there are two common faults that can occur: shifts in mean, and shifts in variance. To simulate a shift in mean, a constant value is added to one of the variables.

To simulate a shift in variance, white Gaussian noise added to one of the variables. The fault is added to the testing region of one variable.

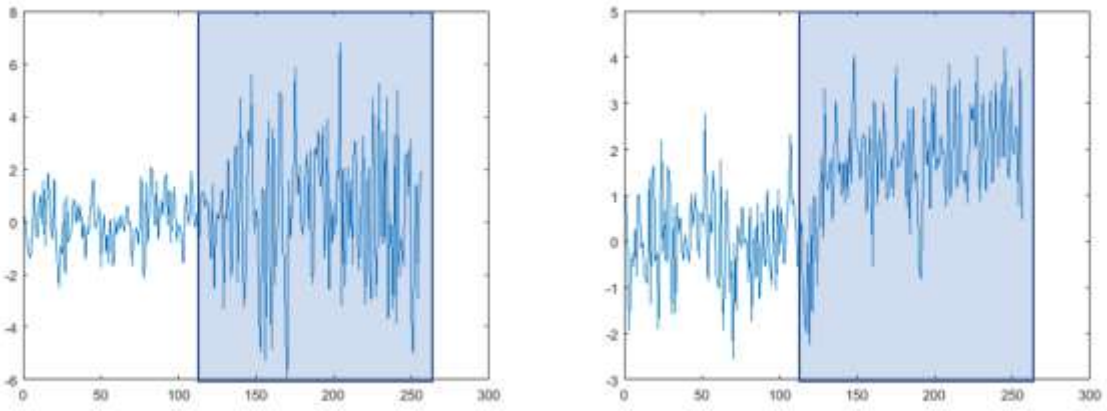

Figure 9: Variable experiencing a shift in variance and shift in mean

2.5.2 Noise generation

In order to mimic real life conditions, noise is added to each variable. In the first two experiments, the noise added is white Gaussian noise with zero mean and certain standard deviation, and the signal to noise ratio (SNR) is calculated in order to quantify this noise. The SNR is the ratio of the standard deviation of the noise free signal, σ_x to the standard deviation of the noise, σ_n [32], given by

$$SNR = \frac{\sigma_x}{\sigma_n} \tag{14}$$

The 2nd type of noise used in work is colored noise which is autocorrelated and had the equation of the form,

$$\eta_k = a * \eta_{k-1} + \eta(0,1) \tag{15}$$

, where a varies between 0 and 1. The addition of noise is essential so that the data mimics real life processes.

The training data is used to train all the methods and generates the predictive and transformation matrices that are then used in the testing data. A Monte Carlo realization of 1000 runs is done for each case, to ensure that the results are consistent.

2.5.3 Interval size

One of the parameters of the IMSPCA method is the choice of how many samples should be included in each generated interval. If the interval size is too large, the method will not be able to capture important features and fault detection will suffer as a result. If the interval size is too small, this defeats the purpose of using intervals to reduce computational complexity. The interval size chosen for this data set was 128 samples per interval, which resulted in the data being compressed to 512 samples.

2.5.4 Choice of depth in multiscale

The choice of multiscale decomposition depth is important in the algorithm: if the depth is too small, the decomposition will not serve its purpose of separating deterministic and stochastic features of the signal and if the depth is too large, important features of the signal will be lost. The depth was then chosen using the heuristic mentioned in Section 2.2.1, and was calculated to be 4 for the interval based methods which had 512 samples, and 8 for the conventional MSPCA, algorithm.

2.6 Results

2.6.1 Case 1: Synthetic linear data set with white noise

In case 1 a linear data set was simulated with white noise. It contained 2 variables that were randomly generated gaussian with a mean and standard deviation of 0 and 2, respectively. The 2nd two variables were linearly related to the first two, as shown below.

$$\begin{aligned} x_1 &= N(\mu_1, \sigma_1) + \eta \\ x_2 &= N(\mu_2, \sigma_2) + \eta \\ x_3 &= x_1 + x_2 + \eta \\ x_4 &= x_1 - x_2 + \eta \end{aligned} \tag{16}$$

The noise that was added was white and gaussian with a standard deviation and mean of 0.5, and the signal to noise ratio was calculated as the standard deviation of the signal divided by the standard deviation of the noise and was equal to 4. Each variable had a sample size of 65,536, and was split into half testing and half training. The CMV method was used to calculate the number of PC retained in each method, and the interval size for the interval-based method was chosen by optimizing the interval length, and was chosen to be 128 samples aggregated per interval. A fault was then added to variable x_1 from sample to 16,385 to sample 49153. In the first simulation shown in Figure 10, it was added as a change in mean in terms of the variable x_1's standard deviation, and in the 2nd simulation it was added a change in variance by adding gaussian white noise with a mean of 0 and standard deviation of 1. The detection rate was plotted for a variety of fault sizes

ranging from +0.3 to +3 change in mean, and +0.3 to +3 change in variance, and are shown in Figure 10 and Figure 12, respectively.

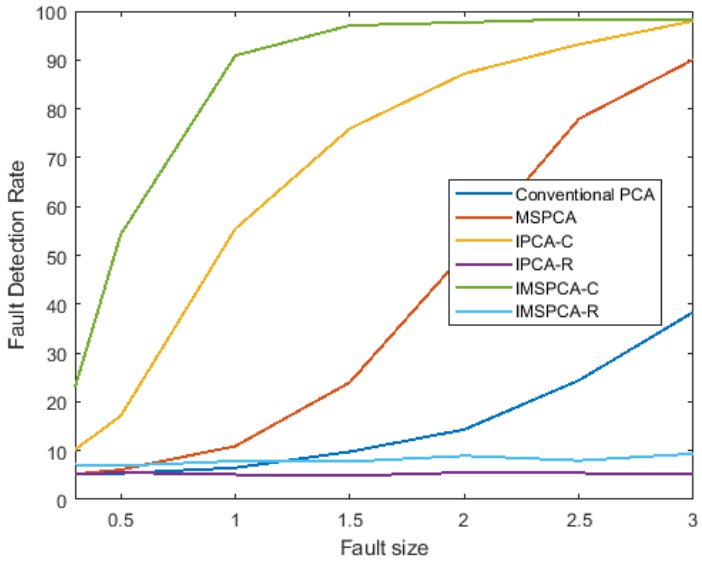

Figure 10 :Fault detection rate versus fault size for a shift in mean of case 1

IMSPCA-C had the highest detection rate at all fault sizes, followed by MSPCA, and then IPCA-C. The radii based interval methods had a constant low detection rate of around 40 and 5% for IMSPCA-R and IPCA-R, respectively. Figure 11 shows the Q statistics for each method.

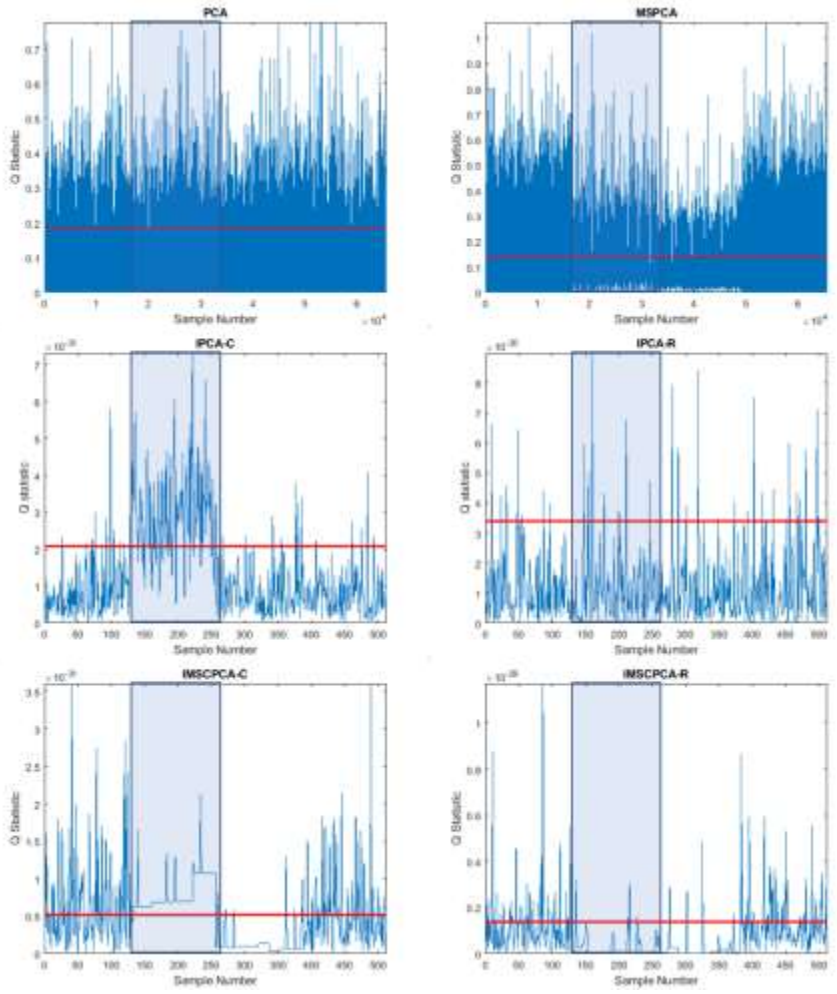

Figure 11: Q statistics of each method for case 1 with a mean shift of 1

As can be seen in Figure 11, the centers based methods are able to detect the shift in mean with much greater accuracy than the radii based methods, which perform quite poorly. In regards to the multiscale decomposition based algorithms, the testing regions contain much less noise than the training regions due to the rules set out in section []. The 95% threshold criteria on the training data means that all of the training data scales are retained, while the samples in the testing region that cross the threshold are set to zero.

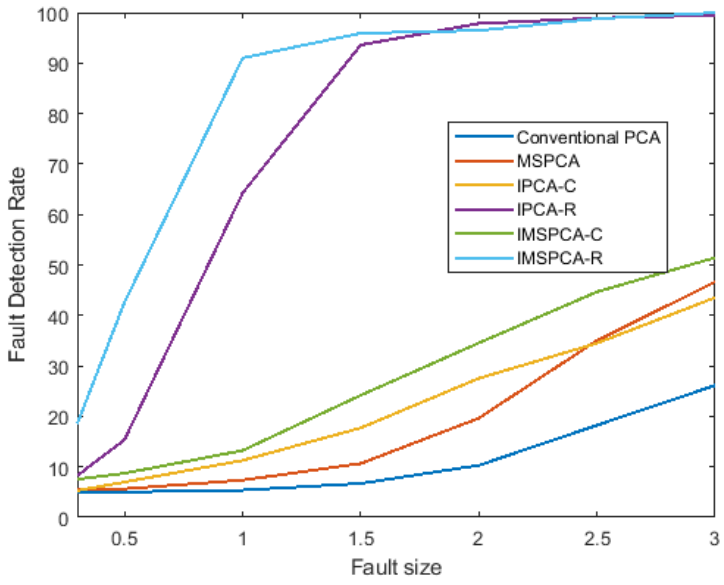

Figure 12: Fault detection rate versus fault size for a shift in variance of case 1

For a change in variance shown in Figure 12 the IMSPCA-R method had the highest detection rate in all cases, followed by IPCA-R and IMSPCA-C. Figure 13 shows the Q statistic for each method.

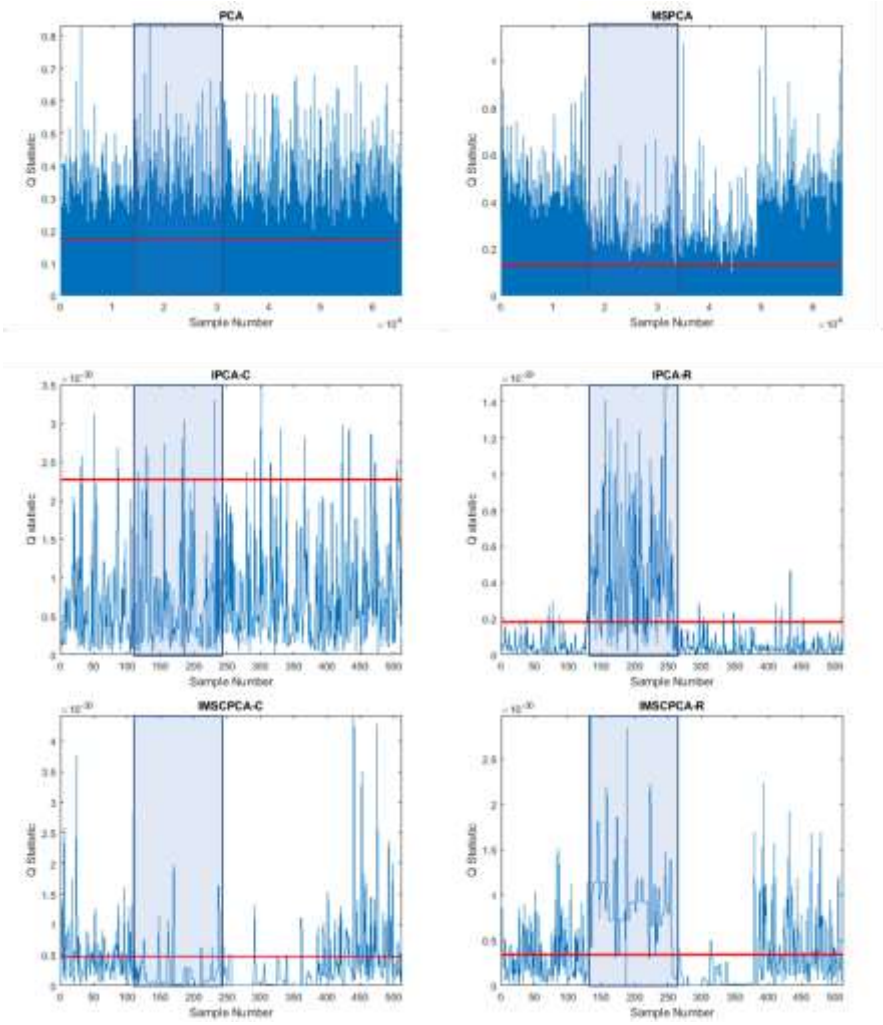

Figure 13:Q statistic of each method for case 1 at a variance shift of 0.5

Figure 13 shows that the radii based methods are much better at detecting shifts in variance than their centers based counterparts, which perform quite poorly in comparison.

2.6.2 Case 2: Synthetic linear data set with autocorrelated noise

In the 2nd case the same data set was simulated as the first case, but the noise added in this case was autocorrelated "pink' noise, with the equation

$$\eta_k = 0.67\eta_{k-1} + \eta(0,1) \tag{17}$$

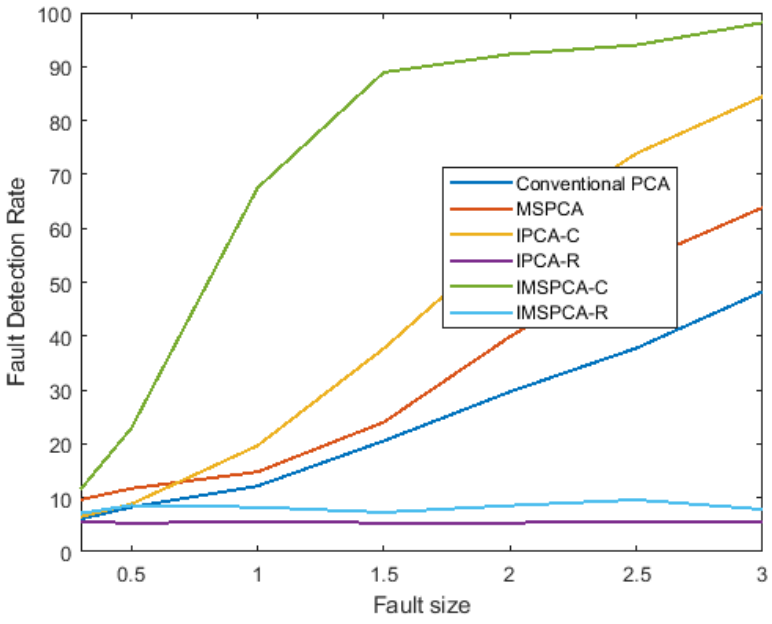

Figure 14: Fault detection rate versus fault size for a shift in mean of case 2

Figure 14 shows the results of case 2 for a shift in mean. IMSPCA-C had the highest detection rate for all the faults, followed by IPCA-C and MSPCA. The Q statistics for each method are shown in Figure 15.

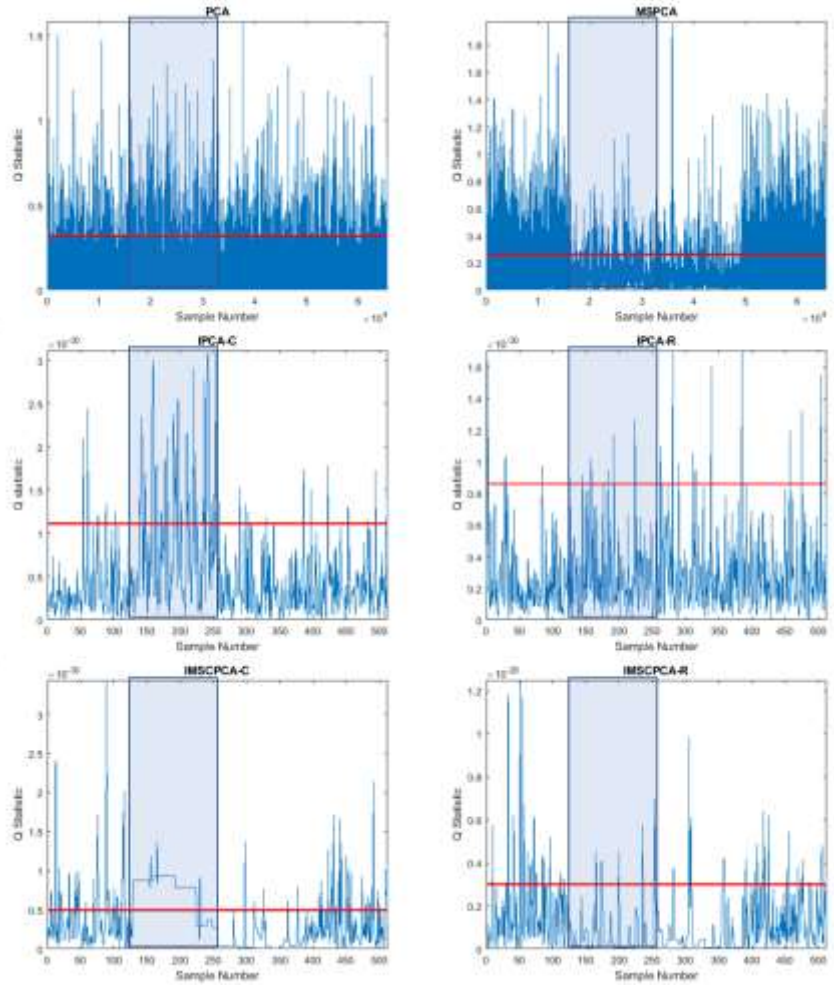

Figure 15: Q statistic of each method for case 2 at a fault size of 1

It can be seen from Figure 15 that the centers based methods show the same trends as case 1 in being better able to detect shifts in mean than the radii based methods.

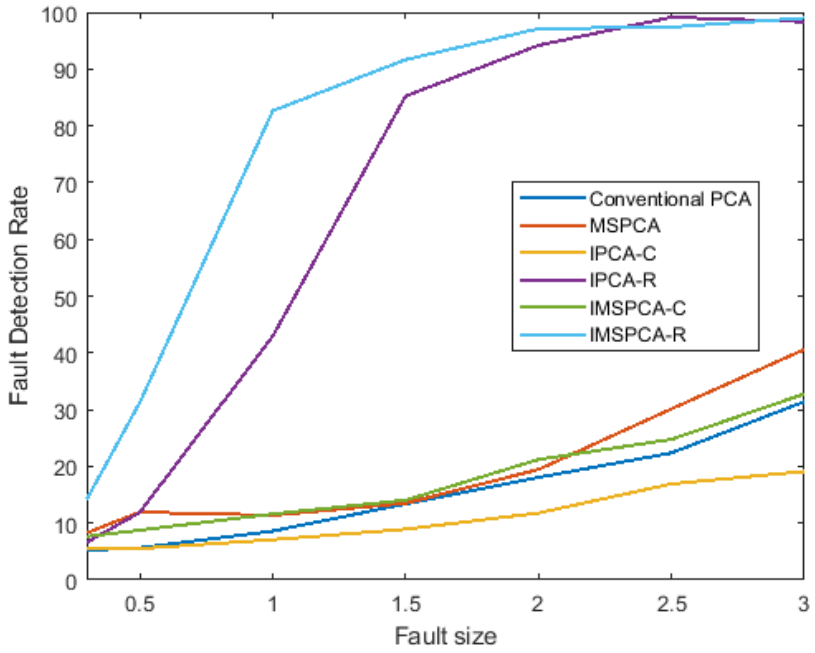

Figure 16: Fault detection rate versus fault size for a shift in variance of case 2

Figure 16 shows the results for a change of variance in case 2. IMSPCA-R had the highest detection rate, followed by IPCA-R. The Q statistics for each case are shown in Figure 17.

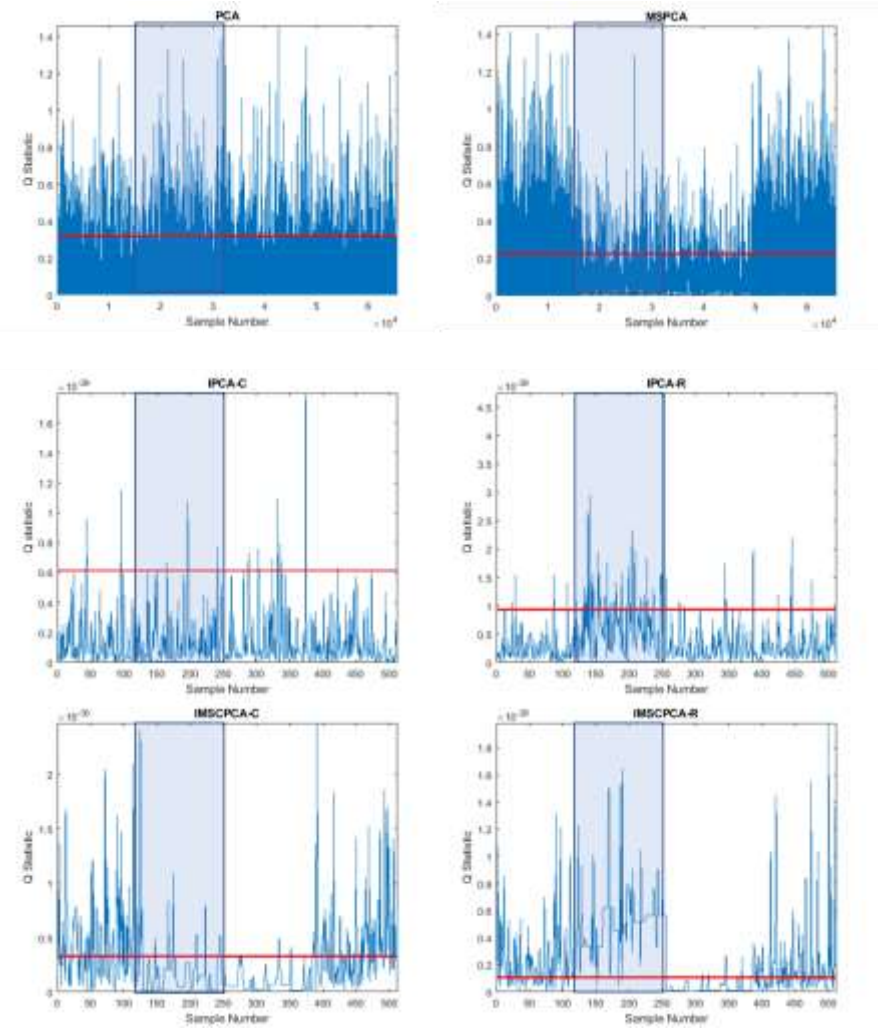

Figure 17:Q statistic of each method for case 2 at a variance shift of 1

The results of the first two cases for a mean shift of +1 and a variance shift of +1 are summarized in Table 1.

Table 1: Summary of detection rates for cases 1 and 2

	Case			
Detection Method	**1 (mean shift)**	**1 (variance shift)**	**2 (mean shift)**	**2 (variance shift)**
PCA	6.50	5.4	12.2	8.6
MSPCA	10.9	7.4	14.8	11.4
IPCA-C	55.4	11.3	19.7	7.1
IPCA-R	5.1	64.2	5.6	42.9
IMSPCA-C	**90.9**	13.2	**67.5**	11.6
IMSPCA-R	7.9	**91.0**	8.3	**82.6**

3. IMPROVING FAULT DETECTION USING NNIMSPCA

3.1 Neural Networks

The extension of PCA to nonlinear methods is a topic of interest when dealing with highly complex data where conventional PCA may fail to detect correlations. Neural networks (NN) are a set of algorithms modeled after neurons in the human brain that seek to discover these correlations in nonlinear data. The data is modeled using fewer components, and the dimensionality of the data is reduced [33]. Neural networks have been used in many applications such as pattern recognition, prediction, optimization, associative memory, and process control [34]. Neural networks have been used specifically in chemical engineering industry in many applications like sensor data analysis, nonlinear process identification, and fault detection [10].

In this work, neural networks are used in the field of process control to improve the accuracy of fault detection in the IMSPCA algorithm for nonlinear data. There are two kinds of NN setup: feedforward and feedback. In a feedforward NN, the signal is passed through successive nodes in the network, starting with the input layer. As the signal moves from one node to the next, it is assigned a specific weight, and the receiving node sums all the weighted signals before processing the output via a function, f, before moving on to the next node. The output in a feedforward network is known beforehand, and the network is assessed based on the difference between the actual output and the desired output. The error between these two is used to optimize the weights until the error is minimized. In a feedback neural network, the weights are kept constant, and the input is changed based on the desired outputs. [35]

3.1.1 Process control using neural networks

Neural networks are a group of nodes that are connected to each other by weights that are optimized during the training period to give the desired output. In a controls problem, the neural network is used to try and reconstruct a set of data, usually a nonlinear dynamic system, using “hidden nodes” or layers within the network. The trained network is then used on testing data, and the deviation of the reconstructed data from the input data is used a measure of the fault in the data. This use of neural networks is termed nonlinear or neural network PCA. In this work, nonlinear PCA is used to modify the IMSPCA algorithm by replacing conventional PCA with nonlinear PCA at all the steps in the algorithm, yielding neural network IMSPCA (NNIMSPCA), and its efficiency in fault detection was assessed using a nonlinear dataset.

3.1.2 Setting up the neural network for nonlinear PCA

Using neural networks in nonlinear PCA, the initial input data, , is connected to a series of hidden layers where the input of each layer, net_j, is a weighted sum of the previous input, and is given by the equation $net_j = \sum_i w_{ji} x_i + b_j$.

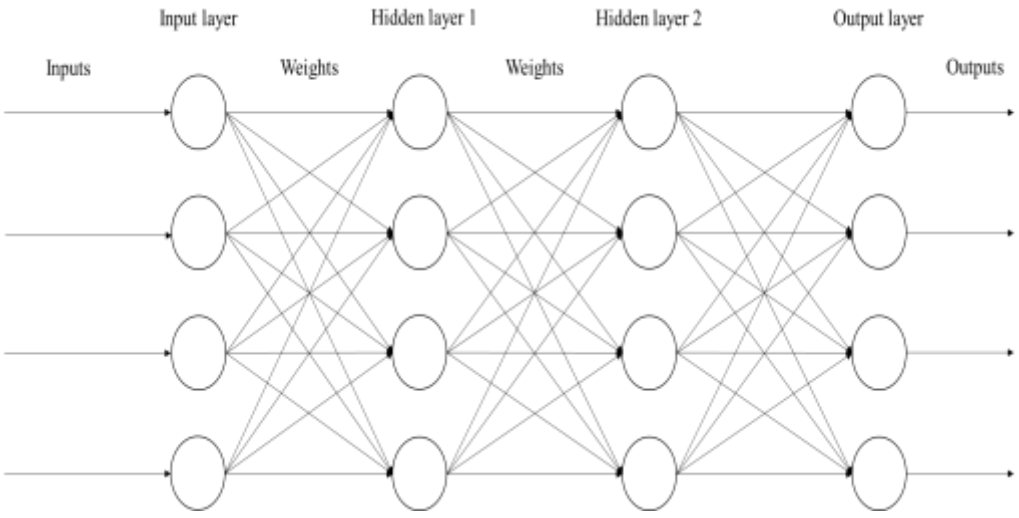

Figure 18: Neural Network with 2 hidden layers

In this equation w_{ji} is a weight assigned to each data point x_i in the input and is used to move it from the *i*th to the *j*th processing element, or node. *b* is the bias of each node and is usually minimized. The data is then mapped into a feature space, the output, using a nonlinear function, $\sigma(x)$, where the most commonly used function being the sigmoid function [36],

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{18}$$

The most commonly used neural network setup is the feed forward network that is trained by back propagation[37], in which the weights are adjusted and input back to the model. It is an iterative method and the weights in the network are adjusted by first calculating the instantaneous error,

$$E = \sum_{q=1}^{n} \left[d_q - y_q \right]^2, \tag{19}$$

where d_q is the expected output of the network at sample q, n is the total number of outputs, and y_q is the actual output of the network, and then changing the weight by the equation,

$$\Delta w_{ij} = -k \frac{dE}{dw_{ij}} \tag{20}$$

where k is a constant of proportionality. The input data is then passed through the network containing the adjusted weights, and this is done until E can no longer be minimized.

3.1.3 Neural network PCA fault detection

In the first step of neural network fitting, the data is trained using a training data set, i.e. a nonfaulty set of data, where the input and outputs are known. The training data set is used as the input and output to the model. This is an iterative step where the weights and bias are optimized in order to give the desired output. After the training is done, the testing data is then passed the network and the output is used to calculate the Q statistic.

The success of the neural network reconstructing the data lies in the presence of a hidden layer, which correspond to principal components in conventional PCA, that contains nonlinear nodes. Without these nonlinear nodes the network would only be able to reconstruct linear functions. These artificial neural networks are called supervised training, because the output for the training network are known [38]. Neural networks are used in order to classify a set of data as faulty or nonfaulty.

3.2 Objectives and Methodology

In this section the parameters used to assess the different algorithms are discussed. The objectives of this work as follows:

1. Develop a combined nonlinear (neural network) IMSPCA algorithm that can detect faults in nonlinear process data with a high accuracy.
2. Perform a comparative study of NNIMSPCA with IMSPCA as well as the conventional PCA, IPCA, and MSPCA algorithms.

3.2.1 Neural network interval multiscale principal component analysis

Due to their computational complexity, neural networks have a disadvantage when dealing with data sets are extremely large, and are not able to process large data sets. They are coupled with interval and multiscale representation in this work in order to solve this problem and yield interval-multiscale neural network principal component analysis (NNIMSPCA), that will be used on nonlinear data.

1. The centers and radii interval matrices $\{X^C, X^R\}$, are generated from the data matrix X. The rest of the steps are done independently on the centers and radii data.
2. Compute the wavelet decomposition of each signal to a depth, J.
3. Collect and group the approximate signals and each detail signal at the different depths from each variable, and compute the covariance matrix for each set.
4. Apply nonlinear PCA, and then use the neural network to compute the Q statistic of the training and testing data sets.

5. For the training data, if any of the Q statistic values cross the threshold, all of the scale is retained.
6. For the testing data, if any of the Q statistic values do not cross the threshold, the sample is zeroed on the original scale
7. Reconstruct the data back into the time domain after applying the above rules, and apply nonlinear PCA.
8. Calculate the Q statistic and assess the fault detection rate and false alarm rate.

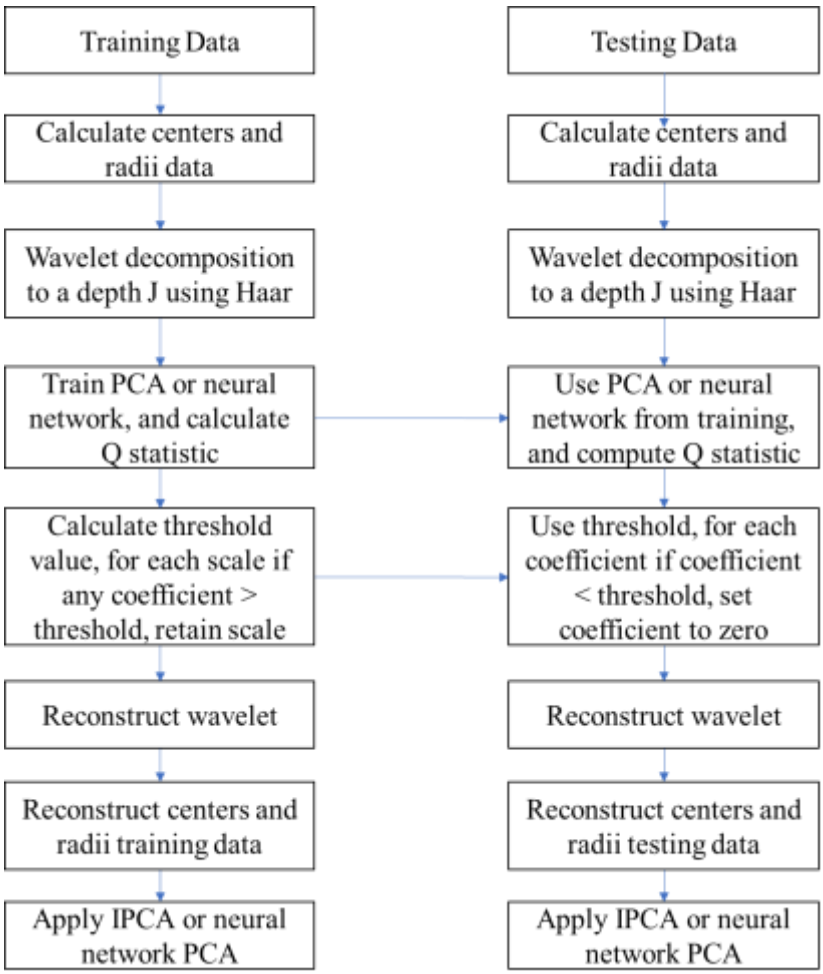

Figure 19: NNIMSPCA schematic diagram

3.2.2 Dataset generation

The data generated for cases 3 and 4 was done similarly to that described in Section [], the main difference being that variables x_3 and x_4 are nonlinearly correlated to the first two variables. In the 3rd and 4th cases, a nonlinear data set was simulated. The first 2 variables were random gaussian with a mean and standard deviation of 0 and 2 respectively, while the last two variables had a nonlinear relationship to the first two, and shown below.

$$\begin{aligned} x_1 &= N(\mu_1, \sigma_1) + \eta \\ x_2 &= N(\mu_2, \sigma_2) + \eta \\ x_3 &= \sin(x_1 x_2) + \eta \\ x_4 &= \cos(x_1 / x_2) + \eta \end{aligned} \tag{21}$$

The sample size was also 65,536, and was split to half training and half testing. For the neural network method, the number of hidden nodes was optimized by simulating it with 3 hidden nodes.

3.3 Results

3.3.1 Case 3: Synthetic nonlinear data set with white noise

Similar to cases 1 and 2, the fault was added to first value with a mean shift ranging from +0.3 to +3 and a variance shift ranging from +.3 to +3 of the standard deviation of x_1, from sample 16,385 to sample 49153.

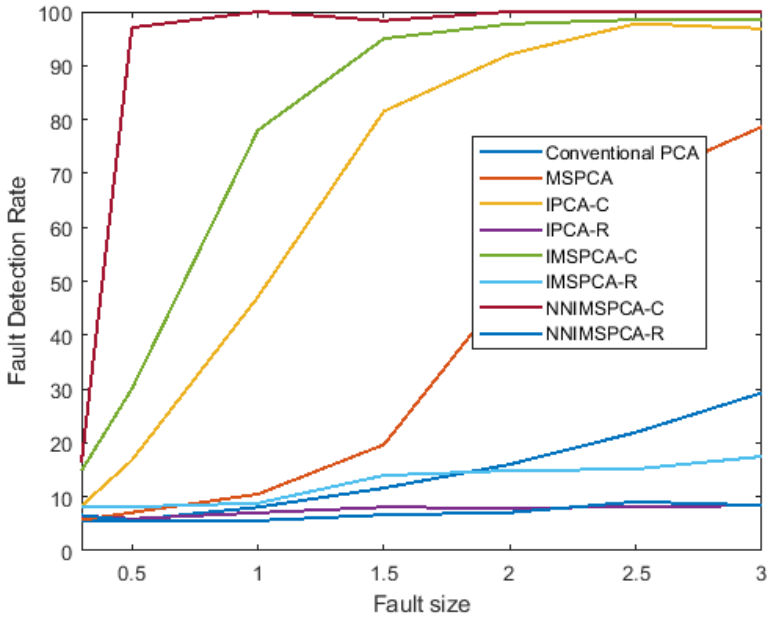

Figure 20: Fault detection rate versus fault size for a shift in mean of case 3

The results for a shift of mean are shown in Figure 20. NNIMSPCA-C performs the best at all fault sizes, followed by IMSPCA-C and IPCA-C. From the graph of the Q statistics shown in Figure 21, it can be seen that the centers based methods perform better than the radii based methods for detecting a shift in mean.

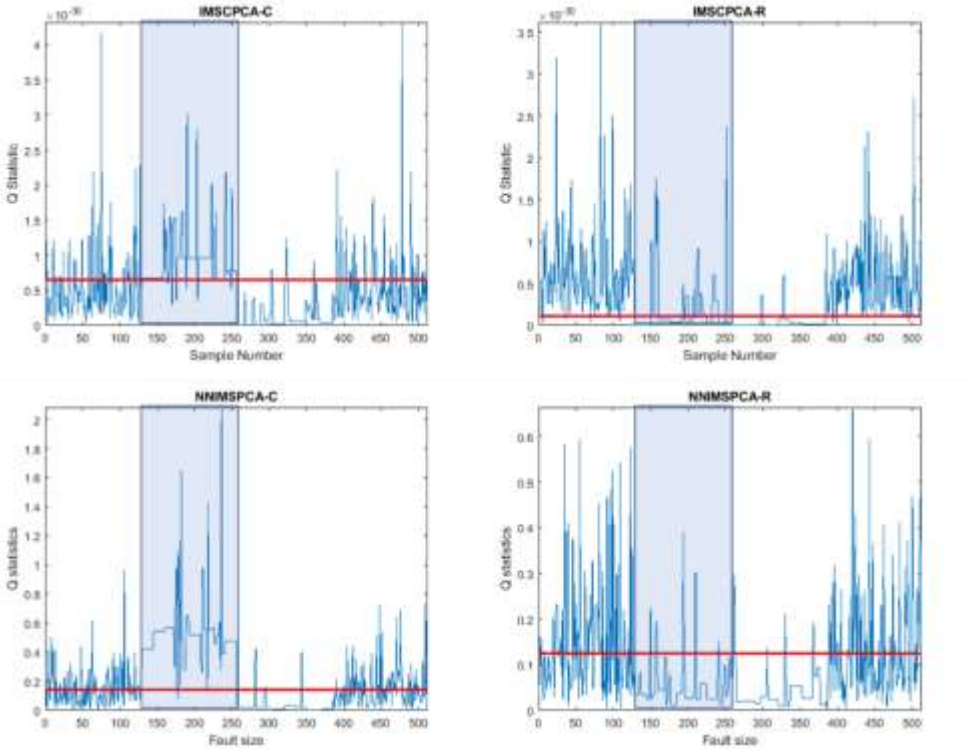

Figure 21: Q statistics for IMSPCA and NNIMSPCA methods for case 3

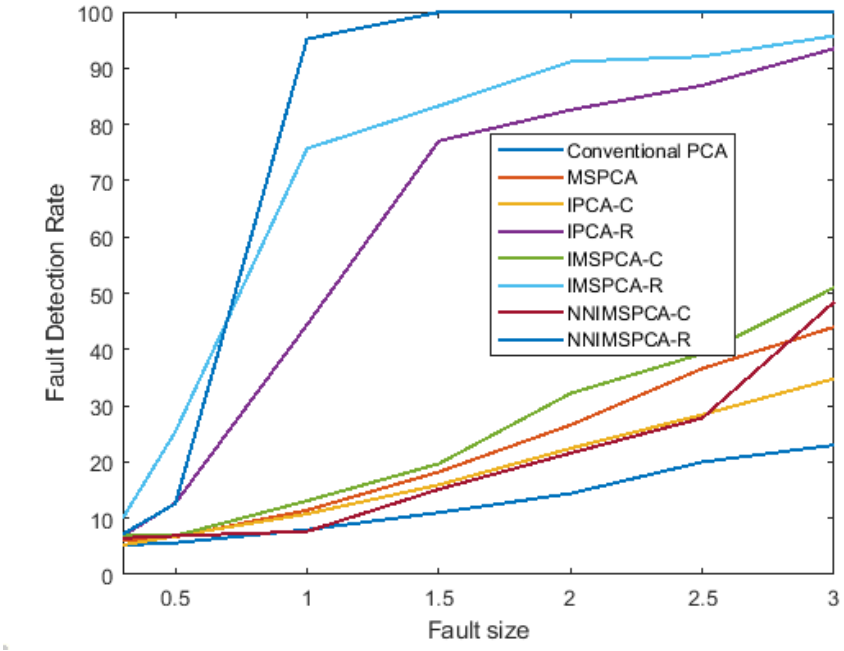

Figure 22: Fault detection rate versus fault size for a shift in variance of case 3

Figure 22 shows the results for a change of variance, and it can be seen that the NNIMSPCA-R method performs the best for most fault size, and fall slightly behind IMSPCA-R at +0.5.

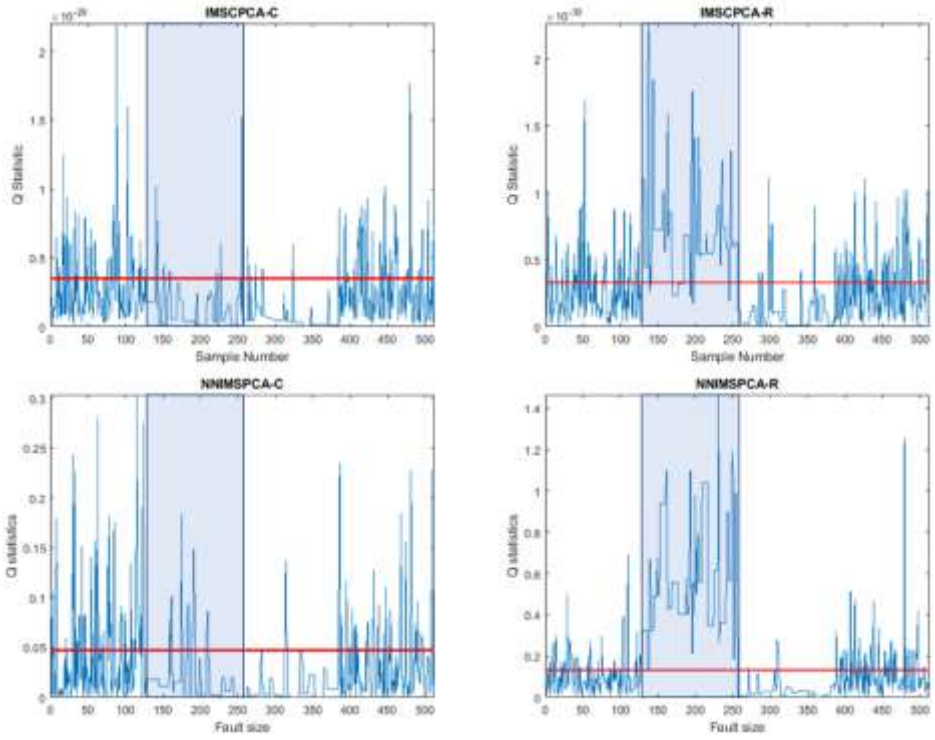

Figure 23:Q statistics for IMSPCA and NNIMSPCA methods for case 3

3.3.2 Case 4: Synthetic nonlinear data set with autocorrelated noise

In the 4th case the same data set was simulated as the 3rd case, but autocorrelated 'pink' noise from case 2 was used instead of white noise.

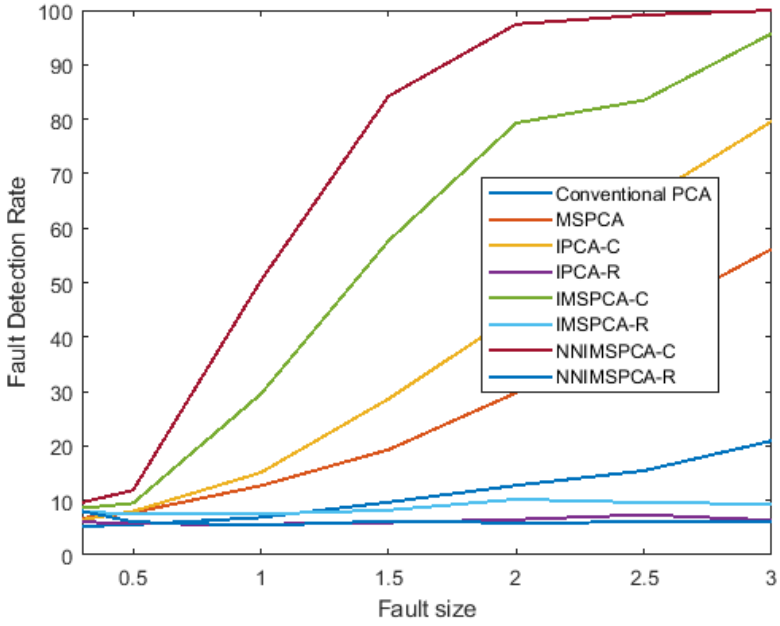

Figure 24: Fault detection rate versus fault size for a shift in mean of case 4

NNIMSPCA-C performs better at all fault sizes, followed by IMSPCA-C and IPCA-C.

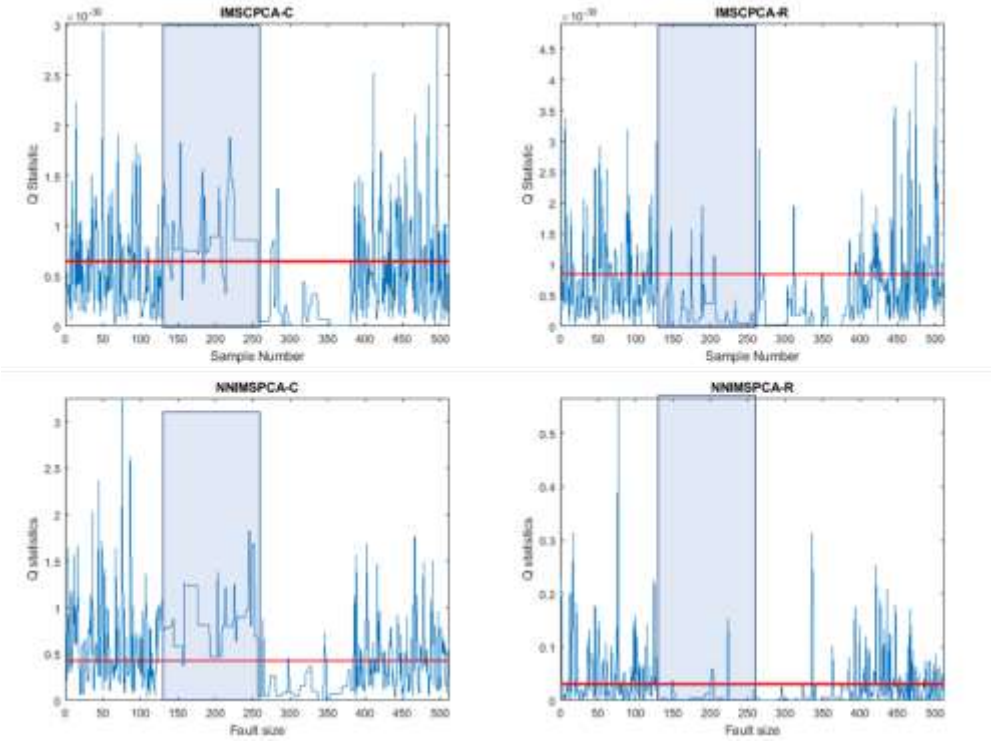

Figure 25: Q statistics for IMSPCA and NNIMSPCA methods for case 4

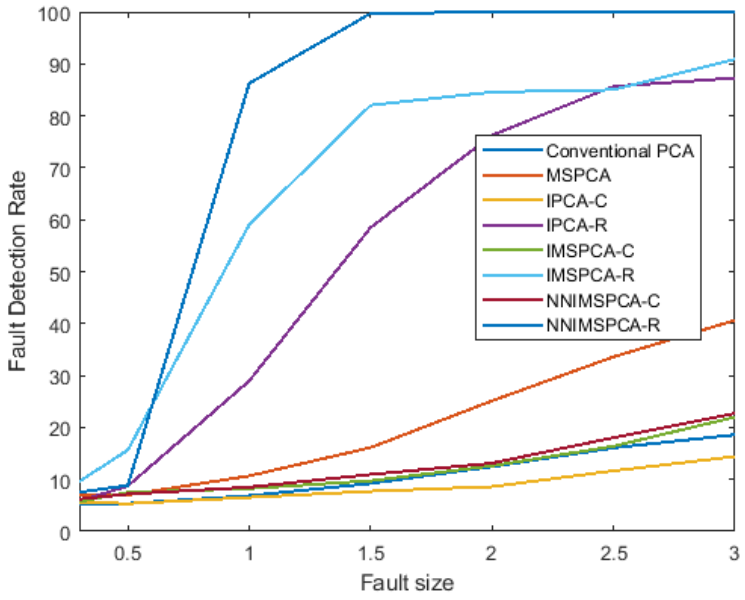

Figure 26: Fault detection rate versus fault size for a shift in variance of case 4

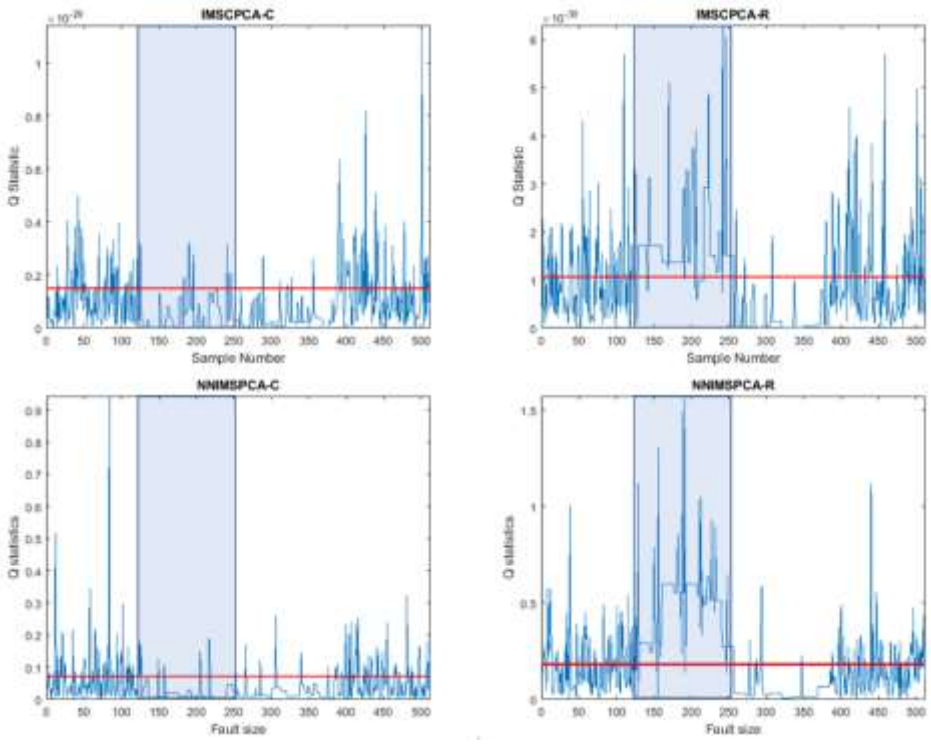

Figure 27: Q statistics for IMSPCA and NNIMSPCA methods for case 4

The results of cases 3 and 4 for a fault size of +1 are summarized in Table 2

Table 2: Summary of results for cases 3 and 4

	Case			
Detection Method	**3 (mean shift)**	**3 (variance shift)**	**4 (mean shift)**	**4 (variance shift)**
PCA	8.1	8.0	6.9	6.8
MSPCA	10.5	11.4	12.7	10.7
IPCA-C	47.1	10.8	15.1	6.5
IPCA-R	7.0	44.4	5.6	29.0
IMSPCA-C	78.0	13.1	29.5	8.2
IMSPCA-R	8.8	75.6	7.5	59.1
NNIMSPCA-C	**100**	7.7	**50.3**	8.5
NNIMSPCA-R	5.6	**95.2**	5.4	**86.3**

4. 100,000 liter penicillin fermentation system

This dataset was acquired from existing data simulated using a realistic reactor design used to produce penicillin in in the pharmaceutical industry, called IndPenSim [39]. The process in nonlinear and 12 variables were input, shown in Table 3.

Table 3: Description of process variables

Variable Index	Description	Units
1	Penicillin concentration	$\frac{g}{L}$
2	Phenylacetic acid concentration	$\frac{mg}{L}$
3	Nitrogen concentration	$\frac{mg}{L}$
4	Viscosity	$\mu - cP$
5	Temperature	K
6	pH	pH
7	Dissolved Oxygen concentration	$\frac{mg}{L}$
8	Weight	kg
9	Off-gas CO2 concentration	ppm
10	Off-gas O2 concentration	ppm
11	Biomass concentration	$\frac{g}{L}$
12	Substrate concentration	$\frac{g}{L}$

The training data obtained from this reactor was divided into training (3000 samples) and testing (3000 samples), and shift of mean varying from +1 to +3 was added

to variable 2 from sample 1 to sample 1500 of the testing data. The FDR was assessed using a 5% FAR value, and the results for each algorithm are shown in Table 4.

Table 4: FDR for various techniques (mean shift)

Mean Shift	PCA	MSPCA	IPCAC	IPCAR	IMSPCAC	IMSPCAR	NNIMSPCAC	NNIMSPCAR
+3	60.1	60.7333	100	8.77	**100**	0.7	**100**	10.3
+2	12.7	13.5	34.3	8.7	84.0	0.7	**100**	10.4
+1	11.26	8.6	32.7	8.7	72.0	0.7	**100**	10.4

The same was also repeated for a variance shift, and white gaussian noise was added to variable 2 from sample 1 to sample 1500 of the testing data. The shift of variance was varied from +1 to +3. The results are shown in Table 5.

Table 5: FDR for various techniques (variance shift)

Variance Shift	PCA	MSPCA	IPCAC	IPCAR	IMSPCAC	IMSPCAR	NNIMSPCAC	NNIMSPCAR
+3.0	38.9	24.9	30.0	99.3	10.0	**100.0**	16.6	**100.0**
+2.0	25.3	7.5	12.0	96.7	6.0	**100.0**	13.1	**100.0**
+1.0	12.3	4.1	6.0	49.3	1.3	76.7	5.6	**84.1**

5. CONCLUSION

Inaccurate fault detection has caused many health, safety, and monetary problems in the past, and is an ever growing and expanding field of work in the chemical engineering industry. Enhanced techniques and algorithms are extremely important to be able to keep up with new processes in the industry. This work aims to contribute to the development of such algorithms that could be applied in order to improve fault detection, in linear and nonlinear processes. In this work, multiscale PCA (MSPCA), which uses wavelet decomposition to separate deterministic and stochastic features in data, is combined with interval PCA, which generates intervals from data that helps reduce the complexity and uncertainty in data, to yield a new method: interval-multiscale PCA (IMSPCA). This algorithm was first tested on linear data, and the results show that IMSPCA has increased fault detection than PCA, MSPCA, and IPCA for linear data. IMSPCA was then modified using neural networks to yield neural network IMSPCA (NNIMSPCA), and the results show that this method has increased fault detection on nonlinear data.

REFERENCES

[1] P. T. Bullemer and I. Nimmo, “A Training Perspective on Abnormal Situation Management: Establishing an Enhanced Learning Environment,” in *AICHE conference on Process Plant Safety*, 1996, pp. 1–9.

[2] J. F. MacGregor and T. Kourti, “Statistical process control of multivariate processes,” *Control Eng. Pract.*, vol. 3, no. 3, pp. 403–414, 1995.

[3] Q. P. He and J. Wang, “Statistical process monitoring as a big data analytics tool for smart manufacturing,” *J. Process Control*, vol. 67, pp. 35–43, Jul. 2018.

[4] P. Russom and T. Org, “BIG DATA ANALYTICS FOURTH QUARTER 2011 TDWI RE SE A RCH,” 2011.

[5] S. Katipamula and M. R. Brambley, “Review article: Methods for fault detection, diagnostics, and prognostics for building systems—a review, part II,” *HVAC R Res.*, vol. 11, no. 2, pp. 169–187, 2005.

[6] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, “A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process,” *J. Process Control*, vol. 22, no. 9, pp. 1567–1581, Oct. 2012.

[7] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemom. Intell. Lab. Syst.*, vol. 2, no. 1–3, pp. 37–52, Aug. 1987.

[8] F. L. Gewers *et al.*, “Principal Component Analysis: A Natural Approach to Data

Exploration," 2018.

[9] W. R. Zwick and W. F. Velicer, "Factors influencing four rules for determining the number of components to retain," *Multivariate Behav. Res.*, vol. 17, no. 2, pp. 253–269, 1982.

[10] M. A. Hussain, "Review of the applications of neural networks in chemical process control - simulation and online implementation," *Artif. Intell. Eng.*, vol. 13, no. 1, pp. 55–68, Jan. 1999.

[11] J. Shlens, "A Tutorial on Principal Component Analysis," 2014.

[12] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4. John Wiley & Sons, Ltd, pp. 433–459, 01-Jul-2010.

[13] R. B. Cattell, "The scree test for the number of factors," *Multivariate Behav. Res.*, vol. 1, no. 2, pp. 245–276, 1966.

[14] S. J. Qin and R. Dunia, "Determining the number of principal components for best reconstruction," *J. Process Control*, vol. 10, no. 2, pp. 245–250, 2000.

[15] R. Cangelosi and A. Goriely, "Component retention in principal component analysis with application to cDNA microarray data," *Biol. Direct*, vol. 2, p. 2, Jan. 2007.

[16] "Advances in Neural Networks – ISNN 2014: 11th International Symposium on ... - Google Books." [Online]. Available:

https://books.google.com.qa/books?id=i46eBQAAQBAJ&pg=PA397&lpg=PA397&dq=cumulative+percentage+variance+commonly+used&source=bl&ots=WEDYCoYCl2&sig=ACfU3U1ICyKIp6GTXxh7wvmQRkPnAKBUTA&hl=en&sa=X&ved=2ahUKEwi1tJngoM7pAhUJnOAKHbCJCkMQ6AEwDHoECAwQAQ#v=onepage&q=cumulative percentage variance commonly used&f=false. [Accessed: 25-May-2020].

[17] P. M., Z. Xu, and Y. Zhao, "Applications of Principal Component Analysis (PCA) in Materials Science," in *Principal Component Analysis - Engineering Applications*, InTech, 2012.

[18] M. Mecozzi, M. Pietroletti, F. Oteri, and R. Di, "Applications of PCA to the Monitoring of Hydrocarbon Content in Marine Sediments by Means of Gas Chromatographic Measurements," in *Principal Component Analysis - Engineering Applications*, InTech, 2012.

[19] F. Castells, P. Laguna, L. Sörnmo, A. Bollmann, and J. M. Roig, "Principal component analysis in ECG signal processing," *EURASIP J. Adv. Signal Process.*, vol. 2007, 2007.

[20] L. G. Cooper and M. Nakanishi, "Standardizing Variables in Multiplicative Choice Models," *J. Consum. Res.*, vol. 10, no. 1, p. 96, 1983.

[21] Z. Chen and Z. Chen, "Evaluation and Comparison of T 2 and Q Statistics for Fault Detection," in *Data-Driven Fault Detection for Industrial Processes*, 2017, pp. 31–42.

[22] L. E. Mujica, J. Rodellar, A. Fernández, and A. Güemes, "Q-statistic and t2-statistic pca-based measures for damage assessment in structures," *Struct. Heal. Monit.*, vol. 10, no. 5, pp. 539–553, 2011.

[23] D. Leung and J. A. Romagnoli, "Chapter 6.4: Fault diagnosis methodologies for process operation," in *Computer Aided Chemical Engineering*, vol. 11, no. C, Elsevier, 2002, pp. 535–556.

[24] B. R. Bakshi, "Multiscale PCA with application to multivariate statistical process monitoring," *AIChE J.*, vol. 44, no. 7, pp. 1596–1610, Jul. 1998.

[25] Z. Shi and Y. Cao, "Application of Haar wavelet method to eigenvalue problems of high order differential equations," *Appl. Math. Model.*, vol. 36, no. 9, pp. 4020–4026, Sep. 2012.

[26] S. Grgic, K. Kers, and M. Grgic, "Image compression using wavelets," in *IEEE International Symposium on Industrial Electronics*, 1999, vol. 1, pp. 99–104.

[27] L. Billard and J. Le-Rademacher, "Principal component analysis for interval data," *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 4, no. 6, pp. 535–540, Nov. 2012.

[28] N. Basha, M. Nounou, and H. Nounou, "Multivariate fault detection and classification using interval principal component analysis," *J. Comput. Sci.*, vol. 27, pp. 1–9, Jul. 2018.

[29] "(19) (PDF) Extensions de l'Analyse en Composantes Principales à des données de type intervalle." [Online]. Available:

https://www.researchgate.net/publication/256822020_Extensions_de_l'Analyse_en_Composantes_Principales_a_des_donnees_de_type_intervalle. [Accessed: 01-Apr-2020].

[30] C. N. Laurol and F. Palumbo2, "Principal Component Analysis for Non-Precise Data."

[31] J. G. Le-Rademacher, "Principal Component Analysis for Interval-Valued and Histogram-Valued Data and Likelihood Functions and Some Maximum Likelihood Estimators for Symbolic Data."

[32] M. Welvaert and Y. Rosseel, "On the definition of signal-to-noise ratio and contrast-to-noise ratio for fMRI data," *PLoS One*, vol. 8, no. 11, Nov. 2013.

[33] M. Scholz, M. Fraunholz, and J. Selbig, "Nonlinear principal component analysis: Neural network models and applications," in *Lecture Notes in Computational Science and Engineering*, 2008, vol. 58, pp. 44–67.

[34] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3. pp. 31–44, Mar-1996.

[35] Y. M. Chiang, L. C. Chang, and F. J. Chang, "Comparison of static-feedforward and dynamic-feedback neural networks for rainfall-runoff modeling," *J. Hydrol.*, vol. 290, no. 3–4, pp. 297–311, May 2004.

[36] V. Venkatasubramanian and K. Chan, "A neural network methodology for process fault diagnosis," *AIChE J.*, vol. 35, no. 12, pp. 1993–2002, Dec. 1989.

[37] D. E. Ruineihart, G. E. Hinton, and R. J. Williams, “Learning Internal Representations Error Propagation,” *Cogn. Sci.*, vol. 1, no. V, pp. 318–362, 1986.

[38] M. A. Kramer, “Nonlinear principal component analysis using autoassociative neural networks,” *AIChE J.*, vol. 37, no. 2, pp. 233–243, Feb. 1991.

[39] S. Goldrick, C. A. Duran-Villalobos, K. Jankauskas, D. Lovett, S. S. Farid, and B. Lennox, “Modern day monitoring and control challenges outlined on an industrial-scale benchmark fermentation process,” *Comput. Chem. Eng.*, vol. 130, p. 106471, Nov. 2019.