Automatic clustering with application to time dependent fault detection in chemical processes

by

P.J. Labuschagne

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

Master of Engineering (Control Engineering)

in the

Department of Chemical Engineering
Faculty of Engineering, the Built Environment and Information Technology

University of Pretoria
Pretoria

1 December 2008

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Automatic clustering with application to time dependent fault detection in chemical processes

P.J. Labuschagne

Automatic clustering with application to time dependent fault detection in chemical processes

Author:	P.J. Labuschagne
Date	1 December 2008
Supervisor:	Carl Sandrock
Department:	Department of Chemical Engineering
	University of Pretoria
Degree:	Master of Engineering (Control Engineering)

Synopsis

Fault detection and diagnosis presents a big challenge within the petrochemical industry. The annual economic impact of unexpected shutdowns is estimated to be $20 billion. Assistive technologies will help with the effective detection and classification of the faults causing these shutdowns.

Clustering analysis presents a form of unsupervised learning which identifies data with similar properties. Various algorithms were used and included hard-partitioning algorithms (K-means and K-medoid) and fuzzy algorithms (Fuzzy C-means, Gustafson-Kessel and Gath-Geva). A novel approach to the clustering problem of time-series data is proposed. It exploits the time dependency of variables (time delays) within a process engineering environment. Before clustering, process lags are identified via signal cross-correlations. From this, a least-squares optimal signal time shift is calculated.

Dimensional reduction techniques are used to visualise the data. Various nonlinear dimensional reduction techniques have been proposed in recent years. These techniques have been shown to outperform their linear counterparts on various artificial data sets including the Swiss roll and helix data sets but have not been widely implemented in a process engineering environment. The algorithms that were used included linear PCA and standard Sammon and fuzzy Sammon mappings.

Time shifting resulted in better clustering accuracy on a synthetic data set based on than traditional clustering techniques based on quantitative criteria (including Partition Coefficient, Classification Entropy, Partition Index, Separation Index, Dunn's Index and Alternative Dunn Index). However, the time shifted clustering results of the Tennessee Eastman process were not as good as the non-shifted data.

Keywords: fault detection, time delay estimation, dimensional reduction, clustering algorithms

Acknowledgements

Firstly, I would like to thank the Lord for all the talents bestowed upon me. To my late father, thank you for all the discussions on the formation of water vapour on the bathroom walls. To my mother, thank you for being the pillar of strength and raising three children. To my study leader, thank you for the all the coffee, wisdom and knowledge that you shared with me. Eats, shoots and leaves!

Lastly, to Nicolien. Thank you for your patience, love and caring ability to get me up and going if times are tough.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

Roman Symbols

$\bar{\mathbf{x}}$	Sample mean
$\hat{\mathbf{x}}$	Mean centred and variance scaled derivatives of the original signal
$\hat{\mathbf{y}}$	Mean centred and variance scaled derivatives of the original delay advanced signal
$\mathbf{I}$	Identity matrix
$\mathbf{v}$	Vector containing the cluster centroids
$\mathbf{x}$	Vector containing an observation from the data
$\mathbf{y}$	Data vectors in reduced space
A	Cluster subset family
A	Coefficients of linear equations in MSTDE optimisation
B	Norm inducing matrix for the fuzzy clustering algorithms
b	Volume of the cluster using the Gustaf-Kessel algorithm
c	Number of cluster centres
D	Distances from data points and cluster centers
d	Distance between two points
E	Error between the higher and lower dimensional inter-point spaces
F	Covariance matrix
J	Clustering objective function
L^*	Vector containing signal lags in MSTDE optimisation

M	Partitioning space
m	Weighting exponent of the fuzzy clustering algorithms
N	Number of data points in the data set
n	Dimensionality of the data set
o	Lower dimensional space
p	Mean square error of the original and recalculated membership functions
R	Correlation matrix
s	Original detected signal in TDE
T	A time series data set
t	Time
U	Cluster partitioning matrix
U^*	Lower dimensional clustering partition matrix
V	Matrix containing the centroids of the clustering algorithm
W	MSTDE weighting function
w	Individual signal weight within W
X	Matrix containing all the data

Greek Symbols

β	Maximum and minimum eigenvalue ratio constraint
χ	Validity index for the clustering problem
Δ	Scatter volume for a cluster in DI
Δ	Statistically valid time delay matrix calculated by the CCF
δ	Point to point cluster separation in the DI
ϵ	Termination tolerance
γ	Covariance matrix weight
κ	Cross correlation time delay
Λ	Diagonal matrix containing all the eigenvalues
λ	Eigenvalue of a matrix
λ_L	Lagrange multiplier
ω	Eigenvector of a matrix

μ	Cluster membership function
Φ	Matrix containing all the eigenvectors
ϕ	Cross correlation value
ψ	Directionality Index
ρ	Correlation index
σ	Standard deviation
τ	Actual time delay calculated by the absolute values of ϕ
θ	Scalar constant for steepest decedent optimisation
ζ	Scaling factor for the time vector in the modified data matrix

Subscripts

ADI	Alternative Dunn's Index
CE	Classification Entropy
DI	Dunn's Index
fc	Fuzzy clustering space
hc	Hard clustering space
i	Current cluster
PC	Partition Coefficient
S	Separation Index
SC	Partition Index
th	Threshold value

CHAPTER 1

Introduction

The role of plant operators has shifted in the last few years from being active participants in the control of a plant, to a broader supervisory role. Modern chemical processing plants produce large amounts of data. This data presents various opportunities for analysis in fault detection, gross error detection etc. Much of this information is not used due to the complexities in extracting useful information. To ease this process, it would be useful if the data could be categorised in terms of the quality of operation, which would enable drilling down further to detect possible causes for concern. Figure 1.1 gives a summary of the primary methods to detect possible faults.

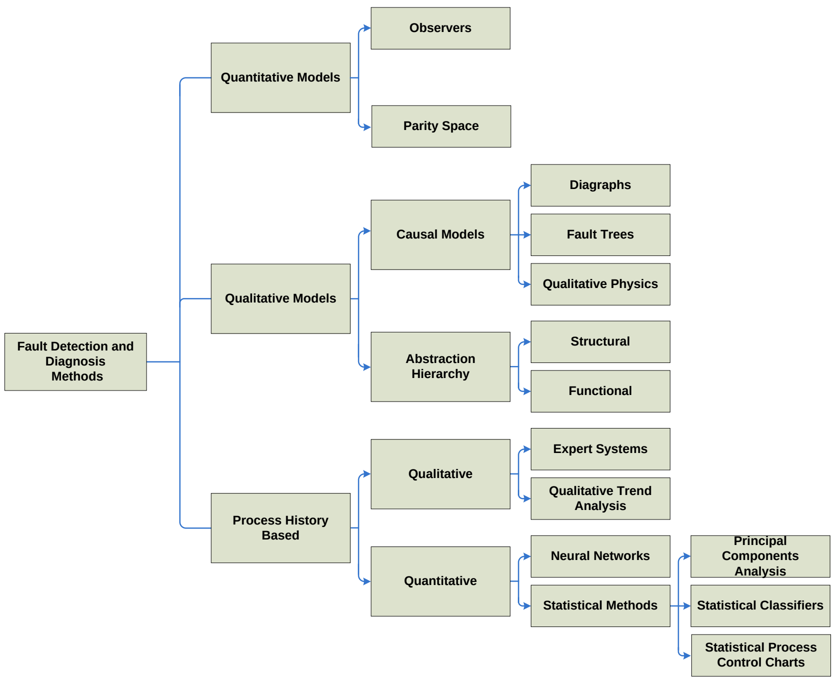


Figure 1.1: Fault detection methods. Adapted from Phillpotts (2007).

Frank (1990) defines a fault as "any kind of malfunction in the actual dynamic system, the plant, that leads to an unacceptable anomaly in the overall system performance. Such malfunctions may occur either in the sensors (instruments), or actuators, or in the components of the process. With respect to the different sectors where the faults can occur, one distinguishes between instrument fault detection (IFD), actuator fault detection (AFD), and component fault detection (CFD)."

1.1 Model based vs. Data based fault detection

Many fault detection techniques detect faults based on differences between actual and expected behaviour. Quantitative models rely on analytical redundancy of explicit models of the system to diagnose the fault. Observer based systems use a set of mathematical observers implemented in a model. Each observer is sensitive to a subset of faults while remaining insensitive to the remaining faults. This makes the diagnosis of multiple faults possible. Parity space relations are generally rearranged variants of input-output models of a plant. Residuals of these relations are used to detect faults. The structure of the model can be used to diagnose or isolate the fault (Phillpotts, 2007)(Frank, 1990). These processes are shown in figure 1.2.

In contrast to model based techniques, process data techniques only require large quantities of process data. This makes data based methods ideally suited to processes that are well instrumented but complex to model, as is characteristic of most modern chemical plants. These data have the potential to provide information for product and process design, monitoring and control. This is especially important in many practical applications, where first-principles modeling of complex "data-rich and knowledge-poor" systems are not possible (Abonyi et al., 2005). Extraction of features or characteristic of the process data can be done in one of two ways (Venkatasubramanian et al., 2003). Qualitative methods include expert systems and trend analysis, unlike quantitative methods can be further classified into neural and statistical methods (Phillpotts, 2007). Data clustering techniques are one of the statistical methods that can be used to detect faulty data.

1.1.1 Statistical Clustering

Humans excel at finding patterns both in everyday life and in data. A set of similar objects can be grouped together. MacKay (2007) gives the following example. *Throughout the years, biologists have found that most objects in the natural world belong to two groups. Those who are green and don't run away and those who are brown and do run away.* The act of grouping objects together is called clustering and in this case two clusters have been identified; plants and animals.

MacKay (2007) lists several motivations for clustering data.

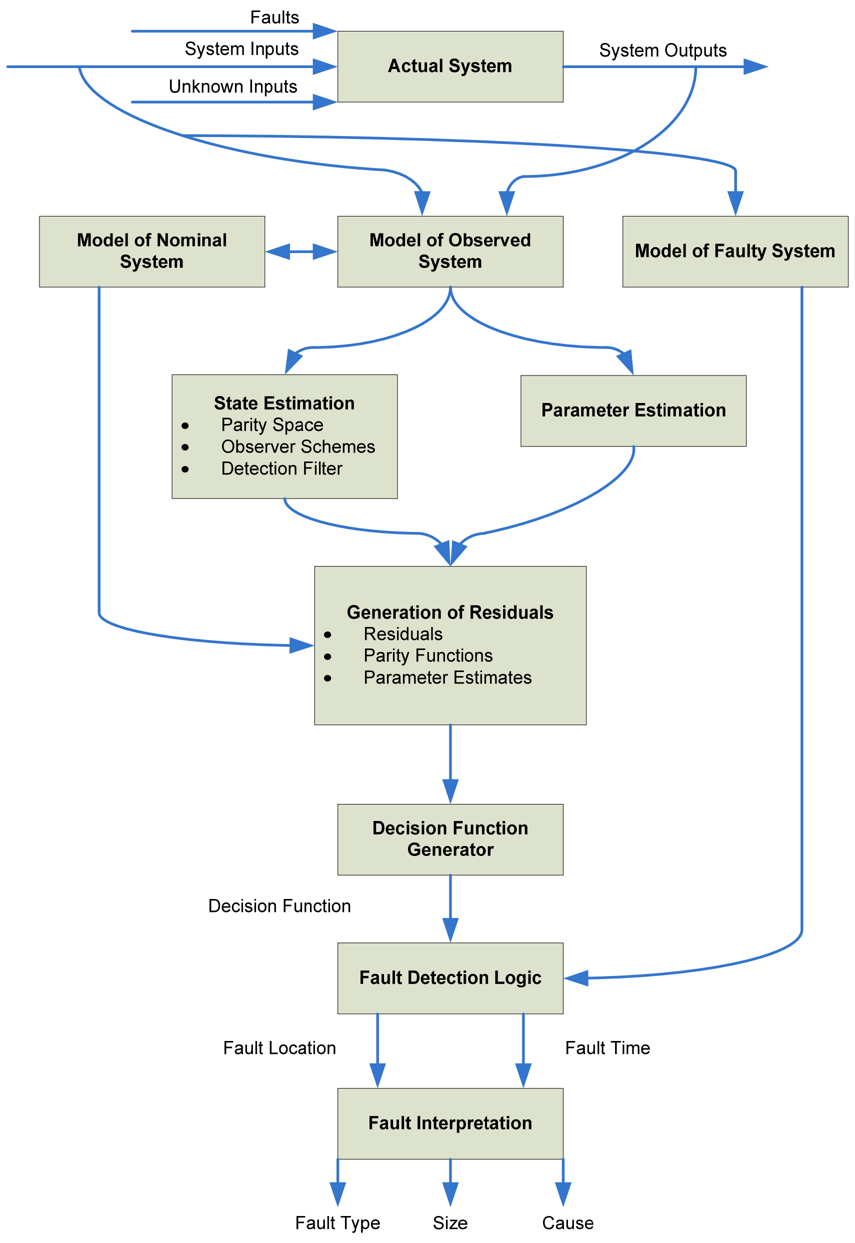


Figure 1.2: General architecture of fault detection and isolation. Adapted from Frank (1990).

1. A good clustering has predictive power. When a biologist encounters something "green" he has never seen before, he can automatically make some form of prediction as to what it is. This is due to the internal model built in previous encounters with such objects. This internal model of plants and animals fills in attributes of this green objects just observed. Will it bite me? Will it graze or sting me? The underlying cluster labels are meaningful and will enable better and more efficient description of the data. This type of clustering is referred to as "mixture density modeling".

2. Clusters can aid communication. The biologist can give directions based on observations made in the field, for example *move towards the fifth tree on the left, turn left towards the third apple tree.* Contrast this with *move towards the green thing with red berries and left towards another green thing with green balls on the sides.* For this reason, clustering is popular in compression algorithms, especially lossy image compression, as the purpose of the algorithm is to convey as few bits as possible as to reconstruct a reasonable reproduction of the image. A common way of doing this is to divide the image up in N small patches and find a close match to a list of K image-templates. After this, a list of labels of matching templates, $k_1, k_2, ..., k_N$, is sent to the picture. The process in which the image-templates are formed is equivalent to finding a set of cluster centres. This type of clustering is referred to as "vector quantization".

3. A failure in the cluster model could highlight important objects that deserve special attention. If a vector quantizer is trained to compress images of brain scans, distortions of these images may be unwanted growths within the brain. If the biologist sees a green object running away, this misfit in his internal model should be accounted for by allowing brown as well as green object to be animals.

Clustering algorithms present an important unsupervised learning problem. According to Güngör & Ünler (2008), data clustering is an NP-complete problem.[1] It is concerned with finding groups in heterogeneous data by minimising some measure of dissimilarity, in other words, the ultimate goal of these algorithms is to find an underlying structure within a given unlabelled data set. Refer to figure 1.3 for a graphical depiction of the process.

Clustering is a challenging field of research, partly because the clustering application is dependant on the specific need of the user. Han & Kamber (2006: p385-386) lists the following criteria with regards to clustering:

[1]NP-complete problems are the hardest problems to solve. NP-complete problems are those problems who's solutions implies solution of all problems in NP.

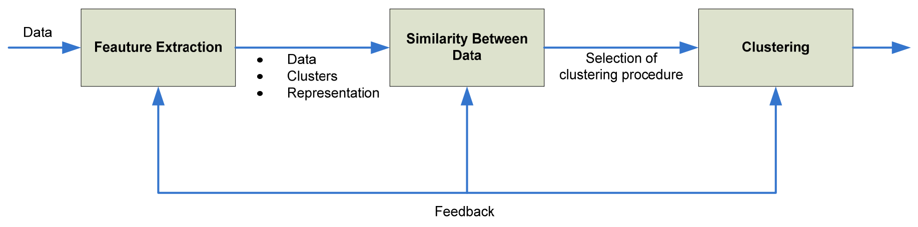


Figure 1.3: The clustering process (adapted from Güngör & Ünler (2008)).

Scalability Normally, a smaller data set produces better clustering results (this is true for many clustering algorithms). Small data sets are however the norm, with some data sets containing millions of entries. For this reason, clustering large data sets may yield biased results. Highly scalable clustering algorithms are needed.

Attribute handling Many algorithms are designed to cluster normal data, i.e. interval based data, normally numerical data. Different applications may require clustering of different types of data like binary, categorical, ordinal data or a combination of these.

Arbitrary shape detection Many clustering algorithms use Euclidean or Manhattan distance norms. These norms tend to identify spherical groupings with similar size and density. Clusters have many shapes and for this reason it is important to develop algorithms that can detect arbitrary shapes.

Requirements of domain knowledge to determine input parameters The majority of clustering algorithms require the user to input a variety of parameters, such as the number of clusters. Clustering results are often very sensitive to these input parameters. Clustering quality can be compromised by the choice of parameters and these are difficult to determine in some cases, especially those with high dimensional data sets.

Noisy data handling Real world data sets contain outliers, unknown, missing or erroneous data. Many clustering algorithms are quite sensitive to this fact and yield poor results.

Insensitivity to the sequence/ordering of input records Some clustering algorithms cannot incorporate new data into the existing clustering structures. These structures would then be recalculated. Also, many algorithms are sensitive to the input sequence/ordering of the data. In other words, if the algorithm processes the data in different sequences, dramatically different results may be obtained. Incremental algorithms that are insensitive to the sequence/ordering of input data are therefore important.

High dimensionality Databases often contain data with various attributes. Many clustering algorithms are adept at handling low dimensional (two or three) data sets. It is easy for a human eye to judge the quality of such a clustering. Clustering in higher dimensions becomes difficult when these data sets are sparse or skewed.

Constrained clustering Real-world applications for clustering algorithms may require constraints. It is a challenging task to find groups of data with good clustering behaviour that satisfy specified constraints.

Interpretability and usability Clustering results must be interpretable, comprehensive and usable. That is, clustering may be tied to specific semantic interpretations and applications. It is important to study how an application goal may influence the selection of clustering features and methods.

There are many clustering algorithms currently available. Some of these algorithms have been used quite extensively in the fields of marketing, biology, earth sciences and more.

1.2 Time Series Clustering

Real-life time series can be taken from physical, social and behavioural science, business, economics, engineering, etc. Abonyi et al. (2005) poses the clustering problem as follows:

> "*Given a time-series, T, find a partitioning of T into c segments that are internally homogeneous. Depending on the application, the goal of the segmentation is to locate stable periods of time, to identify change points, or to simply compress the original time-series into a more compact representation.*"

Although there is a large number of clustering methods and applications, only a few applications have been reported that cluster multivariate time-series data. Singhal & Seborg (2005) summarises some of the most promising work with respect to time-series clustering:

General probabilistic models The authors proposed clustering sequential data using these models, but their approaches were restricted to the use of univariate data. Some work included clustering sequential data by using polynomial regression models but it has limited application to industrial time series data due to the non-linear nature of these data sets.

Probabilistic approach and expectation maximisation algorithms This method involved the estimation of the probability distributions of the steady-states of a system in the multidimensional space. However, this approach is difficult to extend to dynamic systems because the process dynamics blur the distinction between different operating regions.

Data unfolding Multivariate time-series data for a simulated fluid catalytic cracking unit were clustered. The data were clustered by unfolding the multivariate data set into a long row vector and then using the individual elements as features. This method requires that each data set contain the same number of observations, otherwise different data sets contain different amounts of features. This limits the application of this method where the durations of different data sets differ.

Conceptual clustering This method generates "conceptual knowledge" about major variables and projects the data to a specific operational state. Principle components of the data are used to represent the dynamic trends. The data sets are clustered using 2-dimensional plots of the first two principle components. This method requires user input and can become tedious for large data sets.

Wavelet analysis Wavelets, an expectation maximisation algorithm and K-means clustering was used to group univariate time-series data sets. The data set was decomposed using the wavelet transform and clustered the resulting wavelets. This approach was promising, but constrained to univariate data sets.

PCA similarity factors The combination of Euclidean distances and PCA similarity factors were used to cluster different modes of operation for a fluidised catalytic cracker unit. The PCA similarity factors were used to determine the similarity of transitions between plant modes.

Hidden-Markov models These are probabilistic models that not only capture the dependencies between variables, but also the serial correlation in the measurements. For this reason they are well suited for modeling multivariate time series data sets. Although this approach is suitable for multivariate time series clustering, building these models requires an assumed probability distribution or vector quantization. In spite of this, it provides a promising approach to the clustering problem.

Some aspect of all the methods above provide a limitation to either the ease of implementation of the method or the usefulness of the results obtained. Keogh et al. (2003) state that clustering streaming data is "meaningless". Singhal & Seborg (2005) agree with this to a certain extent and states that "the transients appear to blur the distinction between operating conditions and result in either too many of too few clusters". Their results are therefore based on steady-state operation periods. At this point, it is important to distinguish between two different types of time clustering methods. In the one case, similar patterns in a single signal are identified while the other identifies similar regions in various signals. This is illustrated in figure 1.4. In this project we focus on the latter.

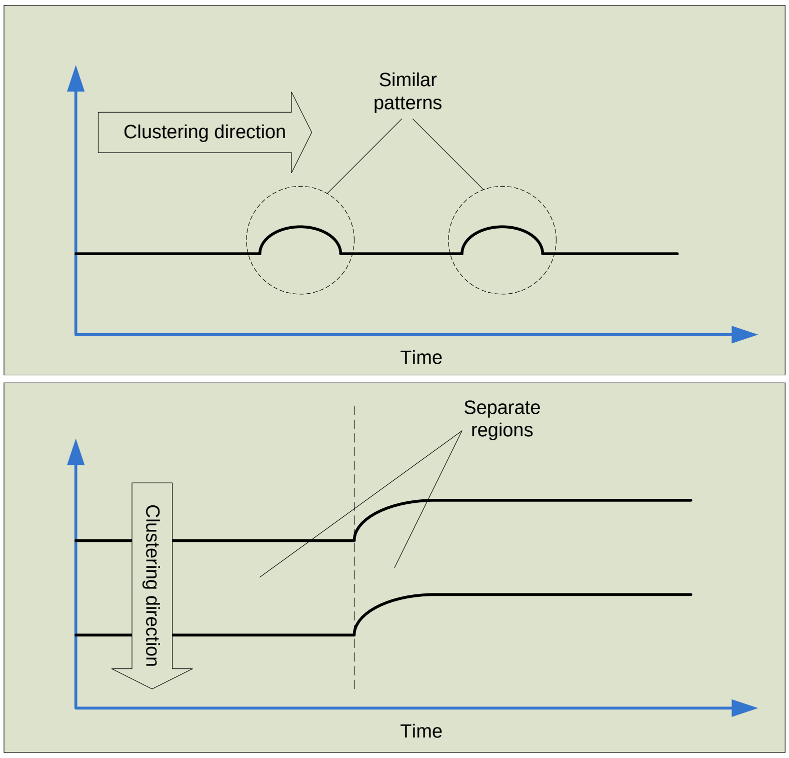


Figure 1.4: Different approaches to clustering time series data. The top approach tries to identify similar patterns within a single signal. The bottom approach tries to identify separate regions within multiple signals.

It is intuitive to consider some form of causal link between process data. Therefore, if a fault originates from some part of the processing plant, it propagates through all the measurements that are linked in some causal way. This fault "event" (or the effect of it) can therefore be detected in these measurements at certain times after the original fault occurred (also referred to as process lags). If these time delays can accurately be determined, and the relevant signals shifted in such a way that the effects coincide in time with the original event, a clustering algorithm may be more effective in distinguishing different operating regions in the data set. The goals of this investigation are therefore to:

- Develop an accurate method to determine the optimal time shifts of signals that are causally linked.
- Cluster the "time-shifted" data set making use of clustering techniques that are easy to implement. This is important as part of the focus of this investigation is to develop assistive-technologies for plant operators, which are not necessarily scientifically inclined.
- Evaluate different dimensional reduction techniques that would assist the user in identifying different periods of operation which would in turn assist in fault detection.

The remaining chapters of the dissertation is structured in as follows.

Data I define the attributes of data sets. We then define the two data sets that will be used in this project: a synthetic data set and the Tennessee Eastman Plant-wide Industrial Control Problem simulation data.

Data Clustering I define relevant literature pertaining to data clustering including various clustering algorithms and validity metrics to evaluate the relevance of each clustering result. We then define dimensional reduction techniques to visualise the resulting higher dimensional clusters.

Time Delay Estimation I define the concept of Multiple Signal Time Delay Estimation (MSTDE). We develop this algorithm from existing cross-correlation and correlation techniques and combine it with recently published statistical thresholding methods to determine the significance of signal correlations. We then show how linear algebra can be used to solve for a single set of time shifts by either minimising the norm of the time shifts (for a rank deficient system) or the norm of the residual (excess rank).

Results: Naive Clustering The clustering algorithms defined in section 3.2 are applied to various data sets. I define benchmarks for perfectly separate clusters and totally random values. I then compare the synthetic data set and the Tennessee Eastman data set to these and quantify the performance of the clustering algorithms on a normal time series data set.

Results: Time Delay Estimation I show that a simple weighting function can increase the performance of the algorithm when complex systems are used. The goal of this section is to calculate a single set of optimum shifts that would increase the performance of the clustering algorithm.

Results: Time Shifted Clustering I combine the results of the previous two chapters. The data is shifted by the optimal calculated shift and then clustered by the K-means and K-medoid clustering algorithms. The optimal number of cluster centers for each data set is calculated as before.

Conclusions, Recommendations and Future Work I draw conclusions based on the results obtained in this project. We also discuss possible future work that would contribute towards refining the techniques used in this project.

CHAPTER 2

Data

We define the attributes of data sets. We then define the two data sets that will be used in this project: a synthetic data set and the Tennessee Eastman Plant-wide Industrial Control Problem simulation data.

Clustering techniques can be applied to data that are quantitative (numerical), qualitative (categorical), or a mixture of both. The data are typically observations of some physical process like persons, houses or documents etc. (Han & Kamber, 2006: 386). Each observation consists of n measured variables, grouped into an n-dimensional column vector $\mathbf{x}_k = [x_{k1}, x_{k2}, ..., x_{kn}]^T, \mathbf{x}_k \in \mathbb{R}_n$. A set of N observations is denoted by $X = \{\mathbf{x}_k | k = 1, 2, ..., N\}$, and is represented as an $n \times N$ matrix (refer to equation 2.1).

$$X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ x_{21} & x_{22} & \dots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nN} \end{pmatrix} \tag{2.1}$$

In pattern recognition terminology, the columns of X are called patterns or objects, the rows are called the features or attributes, and X is called the pattern matrix. In this work, X is often referred to simply as the data matrix. The meaning of the columns and rows of X depends on the context. In medical diagnosis, for instance, the columns of X may represent patients and the rows could be symptoms or laboratory measurements. When clustering is applied to the modelling and identification of dynamic systems, the columns of may X contain samples of time signals and the rows could be physical variables observed in the system (position, velocity, temperature, etc.). In order to capture the system's dynamic behaviour, past values of the variables are typically included in X. Clustering is used to find relationships between independent system variables and future values of dependent variables. However, relationships revealed by clustering are merely casual associations between the data vectors, and as such do not constitute a predictive model of the given system. Additional steps are needed to obtain such a model.

Various synthetic data sets exist in literature. These sets are used to benchmark various clustering and dimensional reduction techniques. Some of these sets are given below (Maszczyk & Duch, 2008) and can also be found at various data repositories including Merz & Murphy (2008).

Parity 8 8-bit parity dataset (8 binary features and 256 vectors).

Heart Disease This set consists 270 samples, each described by 13 attributes. 150 cases belongs to group "absence" and 120 to "presence" of heart "disease".

Wisconsin breast cancer This set contains 699 samples collected from patients. Among them, 458 biopsies are from patients labeled as "benign", and 241 are labeled as "malignant".

Leukaemia This set consists of micro-array gene expressions for two types of leukaemia (ALL and AML), with a total of 47 ALL and 25 AML samples measured with 7129 probes. Visualization is based on 100 best features from simple feature ranking using FDA index.

For the purposes of this project, we are concerned with time-series data that is typically found on chemical processing plants. The remaining sections describe the data sets that were used in this project.

2.1 Synthetic Data Set

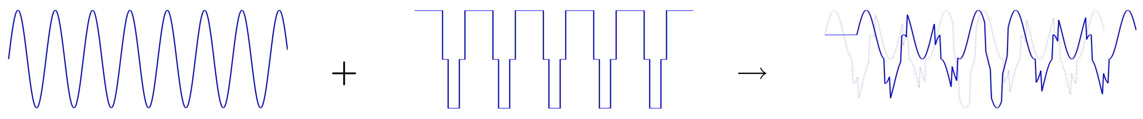
Bauer & Thornhill (2008) state that an oscillatory time series' time delay is periodic with the same frequency as the oscillation and this may cause ambiguities because of phase wrapping. The synthetic data set is constructed to provide a signal which will pass all the relevant criteria with regard to maximum correlation threshold and the directionality index threshold defined by Bauer & Thornhill (2008) (and discussed in chapter 4).

The data set contains 9 signals, paired into 3 groups. Each of the 3 groups have co-prime frequencies. Each of the 3 sets within the data set has an associated time delay. Table 2.1 shows the properties of the different signals.

To construct the signal, the sine wave shown in figure 2.1(a) (with the properties described in table 2.1) was added to a repeating sequence stair as shown in figure 2.1(b), to insure asymmetric waves. The values of this function were $[1, 1, 1, 1, 1, 0, -1, -1, 0]$. This resulted in a signal as shown in figure 2.1(c). Although there are some asymmetrical features on this signal, we still expect some cyclical behaviour.

Table 2.1: Synthetic signal properties.

		Frequency (Hz)	Time Delay
Set 1	Wave 1	0,03	0
	Wave 2	0,03	10
	Wave 3	0,03	5
Set 2	Wave 1	0,011	0
	Wave 2	0,011	12
	Wave 3	0,011	7
Set 3	Wave 1	0,017	0
	Wave 2	0,017	13
	Wave 3	0,017	9



(a) Sine Wave (b) Repeating Stair (c) Combined Signal

Figure 2.1: How the synthetic signals are constructed. Note that the combined signal shows an example of an additional time delayed signal.

2.2 Tennessee Eastman Process

The Tennessee Eastman Plant-wide Industrial Control Problem (TE process) was proposed to test new control and optimisation strategies for continuous chemical processes (Downs & Vogel, 1993). Figure 2.2 shows the flow diagram for the TE process. This diagram includes the primary control loops for the process.

The process consists of the following:

- The coordination of four unit operations including
  - An exothermic two-phase reactor which produces two products from four reactants. Also present in the system are an inert and a byproduct making a total of eight components. For the purposes of the problem, they are named as A, B, C, D, E, F, G and H. The reactions are:

$$\begin{aligned} A(g) + C(g) + D(g) \rightarrow & G(l), \text{ Product 1,} \\ A(g) + C(g) + E(g) \rightarrow & H(l), \text{ Product 2,} \\ A(g) + E(g) \rightarrow & F(l), \text{ Byproduct,} \\ 3D(g) \rightarrow & 2F(l), \text{ Byproduct.} \end{aligned}$$

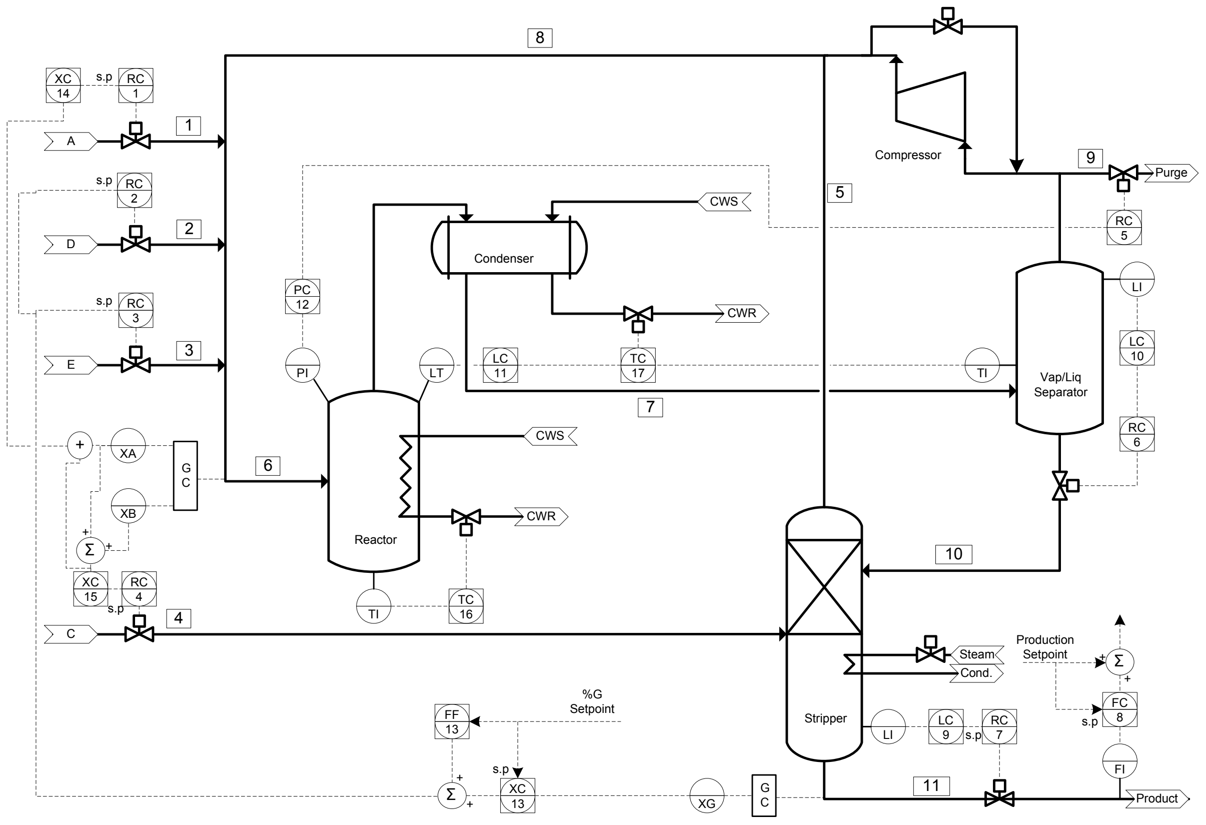


Figure 2.2: Flow diagram of the Tennessee Eastman Plant-wide Industrial Control Problem. Adapted from Ricker (1996).

All the reactions are irreversible and exothermic.

  – A flash separator,
  – A reboiled stripper and
  – A recycle compressor.

- There are 41 measurements with added noise
- There are 12 manipulated variables. These include 11 valves and the reactor agitation speed. Because the agitation speed stays constant for the purposes of this project, it is ignored.
- There are 19 composition measurements, which are sampled at two different rates and include pure time delay.

We recommend the papers presented by Downs & Vogel (1993) and Ricker (1996) for more information on the processes and control layouts. For the purposes of this project, only the simulation data is required to test the various algorithms. Downs & Vogel (1993) provides table 2.2, which contains all the possible process disturbances for this simulation model. Ricker (1996) developed the control strategy and accompanying code.

Table 2.2: Various control challenges within the Tennessee Eastman process.

Variable Number	Process variable	Type
IDV(1)	A/C feed ratio, B composition constant (stream 4)	Step
IDV(2)	B composition, A/C ratio constant (stream 4)	Step
IDV(3)	D feed temperature (stream 2)	Step
IDV(4)	Reactor cooling water inlet temperature	Step
IDV(5)	Condenser cooling water inlet temperature	Step
IDV(6)	A feed loss (stream 1)	Step
IDV(7)	C header pressure loss - reduced availability (stream 4)	Step
IDV(8)	A, B, C feed composition (stream 4)	Random variation
IDV(9)	D feed temperature (stream 2)	Random variation
IDV(10)	C feed temperature (stream 4)	Random variation
IDV(11)	Reactor cooling water inlet temperature	Random variation
IDV(12)	Condenser cooling water inlet temperature	Random variation
IDV(13)	Reaction kinetics	Slow drift
IDV(14)	Reactor cooling water valve	Sticking
IDV(15)	Condenser cooling water valve	Sticking

For the purposes of this project, the IDV(3), IDV(13) and IDV(14) data sets were used. These data sets are concerned with temperature disturbances and should test the capabilities of the algorithms well. Also, they were combined to form one continuous data set with three operating regions. The actual data contained in these sets were obtained from Ricker (2008). Table 2.3 shows all the data present in the data set.

Table 2.3: Manipulated and measured variables for the Tennessee Eastman process. Adapted form Downs & Vogel (1993).

Variable name	Variable Number	Base Case value (%)	Low Limit	High Limit	Units
D feed flow (Stream 2)	XMV (1)	63,053	0	5 811	kgh^{-1}
E feed flow (stream 3)	XMV (2)	53,980	0	8 354	kgh^{-1}
A feed flow (stream 1)	XMV (3)	24,644	0	1,017	$kscmh^{-1}$
A and C feed flow (stream 4)	XMV (4)	61,302	0	15,25	$kscmh^{-1}$
Compressor recycle valve	XMV (5)	22,210	0	100	%
Purge valve (stream 9)	XMV (6)	40,064	0	100	%
Separator pot liquid flow (stream 10)	XMV (7)	38,100	0	65,71	m^3h^{-1}
Stripper liquid product flow (stream 11)	XMV (8)	46,534	0	49,10	m^3h^{-1}
Stripper steam valve	XMV (9)	47,446	0	100	%
Reactor cooling water flow	XMV (10)	41,106	0	227,1	m^3h^{-1}
Condenser cooling water flow	XMV (11)	18,114	0	272,6	m^3h^{-1}
A feed (stream 1)	XMEAS (1)	0,250 52	–	–	$kscmh^{-1}$
D feed (stream 2)	XMEAS (2)	3 664,0	–	–	kgh^{-1}
E feed (stream 3)	XMEAS (3)	4 509,3	–	–	kgh^{-1}
A and C feed (stream 4)	XMEAS (4)	9,347 7	–	–	$kscmh^{-1}$
Recycle flow (Stream 8)	XMEAS (5)	26,902	–	–	$kscmh^{-1}$
Reactor feed rate (stream 6)	XMEAS (6)	42,339	–	–	$kscmh^{-1}$
Reactor pressure	XMEAS (7)	2 705,0	–	–	kPag
Reactor level	XMEAS (8)	75,000	–	–	%
Reactor temperature	XMEAS (9)	120,40	–	–	℃
Purge rate (stream 9)	XMEAS (10)	0,337 12	–	–	$kscmh^{-1}$
Product separator temperature	XMEAS (11)	80,109	–	–	℃
Product separator level	XMEAS (12)	50,000	–	–	%
Product separator pressure	XMEAS (13)	2 633,7	–	–	kPag
Product separator underflow (stream 10)	XMEAS (14)	25,160	–	–	m^3h^{-1}
Stripper level	XMEAS (15)	50,000	–	–	%
Stripper pressure	XMEAS (16)	3 102,2	–	–	kPag
Stripper underflow (stream 11)	XMEAS (17)	22,949	–	–	m^3h^{-1}
Stripper temperature	XMEAS (18)	65,731	–	–	℃
Stripper steam flow	XMEAS (19)	230,31	–	–	kgh^{-1}
Compressor work	XMEAS (20)	341,43	–	–	kW
Reactor cooling water outlet temperature	XMEAS (21)	94,599	–	–	℃
Separator cooling water outlet temperature	XMEAS (22)	94,599	–	–	℃

CHAPTER 3

Data Clustering

We define relevant literature pertaining to data clustering including various clustering algorithms and validity metrics to evaluate the relevance of each clustering result. We then define dimensional reduction techniques to visualise the resulting higher dimensional clusters.

3.1 Cluster Partition

According to Balasko et al. (2005), clusters can be seen as subsets of the original data set. These clusters can be classified as hard (crisp) or fuzzy. Hard clustering techniques are quite common and are based on classical set theory which states that an object either belongs to a specific cluster or not. This results in a number of mutually exclusive subsets of the original data set X. Fuzzy clustering techniques on the other hand allow objects to belong to different clusters simultaneously with various degrees of membership. This idea applies intuitively to clustering processes, as objects seldom belong to only one cluster (Filippone et al., 2008). It allows objects on the boundary between different classes to belong all of these classes. This fuzzy membership can be indicated by a membership degree between 0 and 1. The structure of the partition matrix, U, is given in equation 4.11 (Wang & Zhang, 2007).

$$U = \begin{pmatrix} \mu_{11} & \mu_{12} & \cdots & \mu_{1c} \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2c} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{N1} & \mu_{N2} & \cdots & \mu_{Nc} \end{pmatrix} \tag{3.1}$$

where c is the number of clusters

Hard Partitioning

The objective of a hard clustering algorithm is to partition a given data set, X, into c clusters. The number of clusters should be known beforehand either through prior knowledge of the data or choosing a trial value.

Hard partitions can be defined as a family of subsets $\{A_i \subset X\}, \leq i \leq c$ using classical sets. Equations 3.2, 3.3 and 3.4 define its properties (Balasko et al., 2005) (Wang & Zhang, 2007).

$$\bigcup_{i=1}^{c} A_i = X \tag{3.2}$$

$$A_i \cap A_j = \emptyset \text{ for } 1 \leq i \neq j \leq c \tag{3.3}$$

$$A_i \neq \emptyset \text{ for } 1 \leq i \leq c \tag{3.4}$$

In short, this means:

- The subsets, A_i, contain all the data in X.
- The subsets are disjoint.
- None of these subsets are empty or contain all the data in X.

It can also be expressed in terms of membership functions and these are shown in equations 3.5, 3.6 and 3.7.

$$\bigvee_{i=1}^{c} \mu_{A_i} = 1 \tag{3.5}$$

$$\mu_{A_i} \wedge \mu_{A_j} = 0 \text{ for } 1 \leq i \neq j \leq c \tag{3.6}$$

$$0 \leq \mu_{A_i} \leq 1 \text{ for } 1 \leq i \leq c \tag{3.7}$$

As already mentioned, hard partitioning results in either a 0 or 1 membership function. This is contained within μ_{A_i} and is a characteristic function of A_i. Balasko et al. (2005) recommends the use of an adapted notation to simplify the original notations, hence μ_i is used instead of μ_{A_i} and μ_{ik} replaces $\mu_i(x_k)$.

A $N \times c$ matrix $U = [\mu_{ik}]$ represents the hard partition if its elements satisfy equations 3.8, 3.9 and 3.10.

$$\mu_{ik} \in \{0, 1\} \text{ for } 1 \leq i \leq N \text{ and } 1 \leq k \leq c \tag{3.8}$$

$$\sum_{k=1}^{c} \mu_{ik} = 1 \text{ for } 1 \leq i \leq N \tag{3.9}$$

$$0 < \sum_{i=1}^{N} \mu_{ik} < N \text{ for } 1 \leq k \leq c \tag{3.10}$$

Definition 3.1 (Hard partitioning space) *Let $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^T$ be a finite set and let $2 \leq c < N$ be an integer. The hard partitioning space for X is the set*

$$M_{hc} = \{U \in \mathbb{R}^{N \times c} | \mu_{ik} \in \{0,1\} \forall i,k; \sum_{k=1}^{c} \mu_{ik} = 1 \forall i; 0 < \sum_{i=1}^{N} \mu_{ik} < N \forall k\} \tag{3.11}$$

Fuzzy Partitioning

In the many cases, the data will not fall precisely within one cluster or another. For this reason, it is useful to define a fuzzy partition. The fuzzy partition space can be seen as a generalised version of the hard partition that it allows μ_{ik} to attain values between 0 and 1 (Wang & Zhang, 2007; Filippone et al., 2008).

An $N \times c$ matrix $U = [\mu_{ik}]$ represents the fuzzy partitions and its conditions are given by similar equations to that of the hard partitioning (Equations 3.12, 3.13 and 3.14.

$$\mu_{ij} \in [0,1] \text{ for } 1 \leq i \leq N \text{ and } 1 \leq k \leq c \tag{3.12}$$

$$\sum_{k=1}^{c} \mu_{ik} = 1 \text{ for } 1 \leq i \leq N \tag{3.13}$$

$$0 < \sum_{i=1}^{N} \mu_{ik} < N \text{ for } 1 \leq k \leq c \tag{3.14}$$

Definition 3.2 (Fuzzy partitioning space) *Let $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^T$ be a finite set and let $2 \leq c < N$ be an integer. The fuzzy partitioning space for X is the set*

$$M_{fc} = \{U \in \mathbb{R}^{N \times c} | \mu_{ik} \in [0,1] \forall i,k; \sum_{k=1}^{c} \mu_{ik} = 1 \forall i; 0 < \sum_{i=1}^{N} \mu_{ik} < N \forall k\} \tag{3.15}$$

Therefore, the i-th column of U contains the values of the membership functions of the i-th fuzzy subset of X. Equation 3.9 constrains the summation of each column to 1 (probabilistic constraints). For this reason the total membership of $\mathbf{x}_k$ in X is one.

3.2 Clustering Algorithms

3.2.1 K-means and K-medoid algorithms

The K-means and medoid algorithms are some of the simplest unsupervised learning algorithms that can be used to cluster a given data set (Kao et al., 2008). According to Wu et al. (2008), the K-means algorithm has been discovered by several researchers over the years including Lloyd (1957,1982), Forgey (1965) and McQueen (1967) among others and refined to include the K-medoid algorithm. Also, Gray & Neuhoff (1998) places the K-means algorithms in context with other hill climbing algorithms[1].

Given N data points in an n dimensional space, the algorithm assigns these data points to c clusters (the number of clusters is determined beforehand by the user).

Richard & Dean (2007: 696) give a rudimentary explanation of how the algorithm works. It starts off with the initiation of c seed points from which the clusters will evolve. The algorithm selects points from the data set and assigns a data point to a cluster if the cluster's centroid is the closest to the data point. Now the cluster centroids are recalculated for clusters losing and gaining data points. This process continues until no more reassignments take place.

Implementation of algorithm

MacKay (2007: 285-286) gives a formal definition of the algorithm. Each cluster is defined by a vector $\mathbf{v}$ which corresponds to the centroids of the cluster. The algorithm is as follows:

1. Initialization: Initialize with randomly selected cluster centroids.

2. Assignment step: Each data point, N, is assigned to the closest cluster centroid. The distances of data points from the centroids are calculated through a distance metric. Equation 3.16 shows an example of such a metric.

$$J(X,V) = \sum_{i=1}^{c} \sum_{\mathbf{x}_k \in A_i} \|\mathbf{x}_k - \mathbf{v}_i\|_2 \tag{3.16}$$

3. Selection: The data points that are closest to the centroid of a particular cluster are included within that cluster.

4. Update Step: The new centroids are calculated using equation 3.17, taking into account the newly added data points from the previous step.

[1]Hill climbing is a mathematical optimisation technique which belongs to the local search family.

$$\mathbf{v}_i = \frac{1}{N_i} \sum_{\mathbf{x}_k \in A_i}^{N_i} \mathbf{x}_k \tag{3.17}$$

5. Repeat the assignment and update steps until these assignments do not change.

Altough the K-means algorithm is computationally efficient (Park & Jun, 2008; Filippone et al., 2008; Wu et al., 2008), it is very sensitive to outliers. For this reason, the K-medoid algorithm is used. Representative objects called medoids are considered instead of centroids (or means). Therefore, the cluster centers are the nearest objects to the mean of data in one cluster (Balasko et al., 2005: 8). It can be represented mathematically by equation 3.18. Park & Jun (2008) goes on to discuss the developments of various implementations of this algorithm.

$$V = \{\mathbf{v}_i \in X | 1 \leq i \leq c\} \tag{3.18}$$

From this point on, the K-means and K-medoid algorithms are considered to be the same, with the exception of sensitivity to outliers and the implementation of the code. There are several limitations when using the K-means group of algorithms (Wu et al., 2008; Filippone et al., 2008). The K-means algorithm is a limiting case of fitting data by a mixture of k Gaussians. These Gaussians have identical, isotropic covariance matrices ($F = \sigma^2\mathbf{I}$). So it will fail whenever the data is not well described by hyper-spheres (i.e. non-convex shaped clusters in the data) considering that soft assignments of data points to mixture components are hardened to allocate each data point to the most likely component. A possible solution is to scale the data before clustering. This "whitens" the data. Also, using distance measures more appropriate to the given data set could have a positive effect on the results.

Wu et al. (2008) suggest pairing the K-means algorithm with one that describes the non-convex clusters. This could include obtaining a large number of clusters by using K-means on the initial data set. These groups are then agglomerated into larger clusters using single link hierarchical clustering, which can detect complex shapes. One of the advantages to this procedure is that it makes the solutions less dependent in the initialization step in the algorithm. Initialization could be problematic due to local minima on the $E(X)$ surface (Filippone et al., 2008; Kao et al., 2008).

The cost of the optimal solution decreases with an increase in the number of clusters (c). This continues until the number of clusters equal the number of data points in the data set. This complicates both the direct comparison of solutions with different numbers of clusters and finding the optimum c.

To improve the solution generated by the K-means algorithm and address some of the issues mentioned above, it can be "kernalised". This retains the linear boundaries between the clusters in the higher dimensional space, but can become non-linear when projected back to the original input space.

Figures 3.1 and 3.2 are examples of these hard clustering techniques. Three cluster centers were used in this clustering process and it is evident from these figures that there are errors in the clusters. This highlights the hard nature of these algorithms. Note that different clusters may be formed every time the algorithm is implemented due to the random seeds that are generated to start the process.

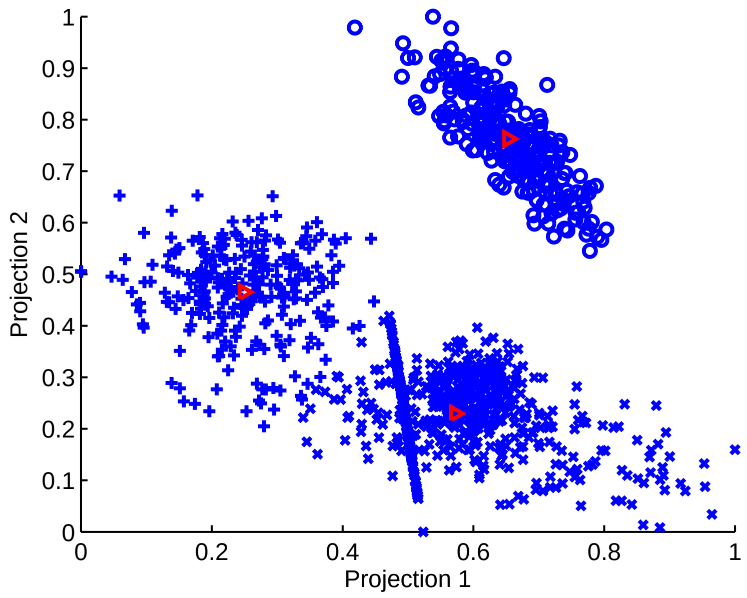


Figure 3.1: Result of the K-means algorithm by using a synthetic overlapping data set.

3.2.2 Fuzzy C-Means algorithm

The Fuzzy C-Means (FCM) algorithm is based on the minimisation of the *C-means functional* objective function to find V. The objective function is defined by equation 3.19 (Filippone et al., 2008; Balasko et al., 2005).

$$J(X; U, V) = \sum_{i=1}^{c} \sum_{k=1}^{N} (\mu_{ik})^m \|\mathbf{x}_k - \mathbf{v}_i\|^2 \tag{3.19}$$

The parameter m controls the fuzziness of the memberships (Filippone et al., 2008). The normal value for this parameter is 2. For higher values of m, the algorithm tends to set all the memberships equal, whereas a value of 1 results in the K-means algorithm discussed in section 3.2.1.

Balasko et al. (2005) notes that equation 3.19 could be considered of a total variance measure of $\mathbf{x}_k$ from $\mathbf{v}_i$.

Minimisation of equation 3.19 is a non-linear optimisation problem which can be solved

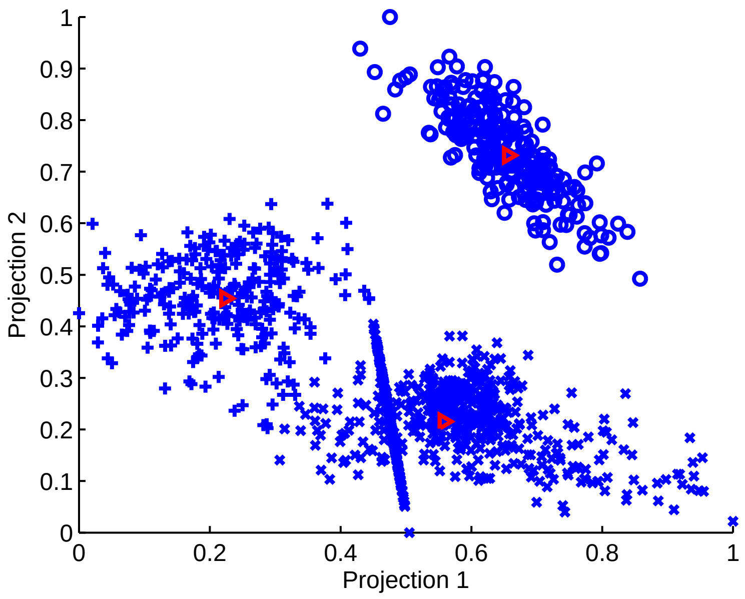


Figure 3.2: Result of the K-medoid algorithm by using a synthetic overlapping data set.

with methods including grouped coordinate minimisation, simulated annealing, genetic algorithms and many more. Balasko et al. (2005) found the most popular method to be a simple Picard iteration through the first-order conditions for the stationary points of equation 3.19. This is known as the FCM algorithm. Abonyi et al. (2002) notes that although the classical FCM algorithm is able to detect groups of data, it is not organised and for this reason it makes interpretation of the model difficult.

Derivation

The stationary points of the C-means functional in equation 3.19 can be determined by adjoining the constraint found in equation 3.13 to J by means of Lagrange multipliers. This is shown in equation 3.20 (Filippone et al., 2008).

$$\bar{J}(X,U,V,\lambda_L) = \sum_{i=1}^{c}\sum_{k=1}^{N}(\mu_{ik})^m D_{ikB}^2 + \sum_{k=1}^{N}\lambda_{L_k}\left(\sum_{i=1}^{c}\mu_{ik} - 1\right) \tag{3.20}$$

where

$$D_{ikB}^2 = \|\mathbf{x}_k - \mathbf{v}_i\|^2 = (\mathbf{x}_k - \mathbf{v})^T B(\mathbf{x}_k - \mathbf{v}_i) \tag{3.21}$$

After this, the derivatives of $\bar{J}$ with respect to U and V are set equal to zero. If $D_{ikB}^2 > 0 \forall i,k$ and $m > 1$, then $(U,V) \in M_{fc} \times \mathbb{R}^{n\times c}$ will minimise equation 3.19 if and only if equations 3.22 and 3.23 hold true.

$$\frac{1}{\mu_{ik}} = \sum_{j=1}^{c}\left(\frac{D_{ikB}}{D_{jkB}}\right)^{\frac{2}{(m-1)}}, \forall 1 \le i \le c, 1 \le k \le N \tag{3.22}$$

$$\mathbf{v}_i = \frac{\displaystyle\sum_{k=1}^{N} (\mu_{ik})^m \mathbf{x}_k}{\displaystyle\sum_{k=1}^{N} (\mu_{ik})^m}, \forall 1 \leq i \leq c \tag{3.23}$$

This solution satisfies the remaining constraints given in equations 3.12 and 3.14. Equation 3.23 gives $\mathbf{v}_i$ as the weighted mean of the data items that belong to a cluster. These weights are merely the membership degrees. The FCM algorithm becomes a simple iteration through equations 3.22 and 3.23.

An important limitation to this algorithm is the fact that it makes use of the standard Euclidean distance norm. This produces hyper-spherical clusters and for this reason it can only detect clusters with the same shape. This is due to the common choice of the norm inducing matrix, $B = I$ (Balasko et al., 2005; Abonyi et al., 2002; da Silva et al., 2008). This matrix can also be chosen as a $n \times n$ diagonal matrix that accounts for different variances in the directions of the coordinate axes of X. This is shown in equation 3.24.

$$B = \begin{pmatrix} \left(\frac{1}{\sigma_1}\right)^2 & 0 & \dots & 0 \\ 0 & \left(\frac{1}{\sigma_2}\right)^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \left(\frac{1}{\sigma_n}\right)^2 \end{pmatrix} \tag{3.24}$$

B can also be defined as the inverse of the $n \times n$ covariance matrix, $B = F^{-1}$. This is shown in equation 3.25. In this instance, B induces the Mahalanobis norm on $\mathbb{R}^n$ (Balasko et al., 2005; Liu & Xu, 2008).

$$F = \frac{1}{N} \sum_{k=1}^{N} (\mathbf{x}_k - \bar{\mathbf{x}})(\mathbf{x}_k - \bar{\mathbf{x}})^T \tag{3.25}$$

Implementation of algorithm

Given a data set X, choose the number of clusters, c, so that $1 \leq c \leq N$, the weighting exponent m, so that $m > 1$ as well as the termination tolerance, $\epsilon > 0$ and the norm inducing matrix B. Then initialise the partition matrix, such that $U^{(0)} \in M_{fc}$ (Balasko et al., 2005) quoting (Bezdek, 1981).

Repeat for $l = 1, 2, \dots$

1. Compute the cluster centers by means of equation 3.26.

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m \mathbf{x}_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \forall 1 \leq i \leq c \tag{3.26}$$

2. Compute the mutual distances using equation 3.21.

3. Update the partition matrix with equation 3.27.

$$\frac{1}{\mu_{ik}^{(l)}} = \sum_{j=1}^{c} \left(\frac{D_{ikB}}{D_{jkB}} \right)^{\frac{2}{(m-1)}}, \forall 1 \leq i \leq c, 1 \leq k \leq N \tag{3.27}$$

4. Steps (1) - (3) are repeated until $\|U^{(l)} - U^{(l-1)}\| < \epsilon$.

Figure 3.3 gives an example of the output from the FCM algorithm. As mentioned earlier, the FCM algorithm can only detect clusters with circular shapes. For this reason, the algorithm struggles to detect the linear group of data at the bottom of the figure. Balasko et al. (2005) notes that the FCM algorithm is a very good initialisation tool for more sensitive algorithms like the Gath-Geva algorithm discussed in section 3.2.4.

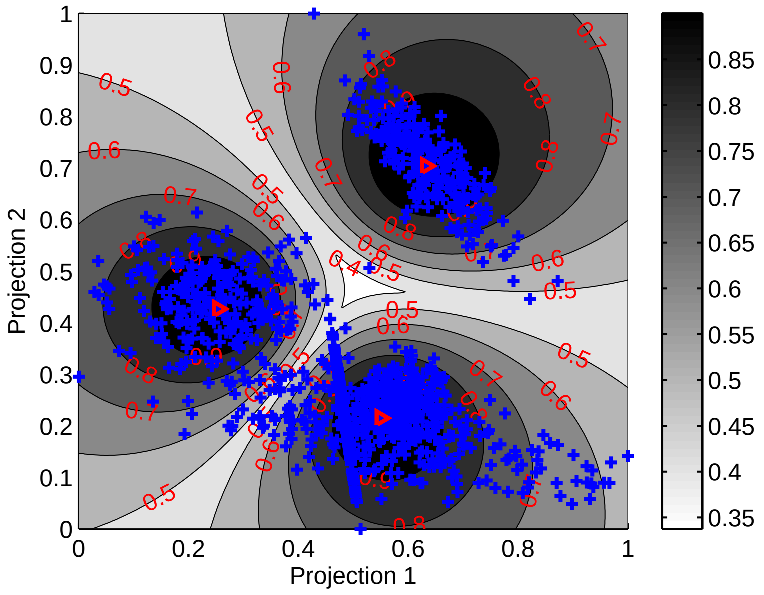


Figure 3.3: Result of the Fuzzy C-Means algorithm by using a synthetic overlapping data set. The gradient lines presented on the figure are linear representations of the partitioning between different clusters.

3.2.3 Gustafson-Kessel algorithm

Following from the FCM algorithm, the Gustafson-Kessel (GK) algorithm applies the same reasoning but employs an adaptive distance norm. This enables the algorithm to detect different geometric shapes in one data set (Balasko et al., 2005; da Silva et al., 2008). Each of the clusters has its own norm-inducing matrix B_i, which yields the inner-product norm contained in equation 3.28.

$$D^2_{ikB_i} = \|\mathbf{x}_k - \mathbf{v}_i\| = (\mathbf{x}_k - \mathbf{v})^T B_i (\mathbf{x}_k - \mathbf{v}_i) \tag{3.28}$$

In this algorithm, the norm-inducing matrices are used as optimization variables in the C-means functional. This allows each cluster to adapt the distance norm to take the topological structure of the data into account.

Derivation

Let $\mathbf{B}$ denote a c-tuple of the norm inducing matrices: $\mathbf{B} = (B_1, B_2, \ldots, B_c)$. The objective function of the GK algortihm is defined in equation 3.29.

$$J(X, U, V, \mathbf{B}) = \sum_{i=1}^{c} \sum_{k=1}^{N} (\mu_{ik})^m D^2_{ikB_i} \tag{3.29}$$

It follows from this equation that if B is kept constant, the fuzzy partition criteria given in equations 3.12, 3.13 and 3.14 can be applied. However, since equation 3.29 is linear in terms of B_i, it cannot be minimised directly with respect to B_i. J can be made as small as possible by merely making B_i as small as possible. For this reason, B_i should be constrained in some way. da Silva et al. (2008) recommend that the determinants of B_i be constrained. Allowing B_i to vary while their determinants are fixed, allows for the shape of the cluster to be optimised while the volume remains constant. This is shown in equation 3.30.

$$\|B_i\| = b_i \tag{3.30}$$

with b_i fixed for each cluster. Equation 3.31 is obtained by using the Lagrange multiplier method. The fuzzy covariance matrix, for the i-th cluster is given in equation 3.32

$$B_i = [b_i \det(F_i)]^{\frac{1}{n}} F_i^{-1} \tag{3.31}$$

$$F_i = \frac{\sum_{k=1}^{N} (\mu_{ik})^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^T}{\sum_{k=1}^{N} (\mu_{ik})^m} \tag{3.32}$$

The combination of equations 3.31,3.32 and equation 3.28 yields a generalised squared Mahalanobis norm between $\mathbf{x}_k$ and the cluster mean $\mathbf{v}_i$ where the covariance is weighted by the membership degrees in U.

Implementation of algorithm

Given a data set X, choose the number of clusters such that $1 < c < N$, the weighting exponent such that $m > 1$, the termination tolerance $\epsilon > 0$ and the norm inducing matrix B. Now, initialise the partition matrix, such that $U^{(0)} \in M_{fc}$ (Balasko et al., 2005).

Repeat for $l = 1, 2, \ldots$

1. Compute the cluster centers by means of equation 3.33.

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m \mathbf{x}_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \forall 1 \leq i \leq c \tag{3.33}$$

2. Compute the cluster covariance matrices by using equation 3.34.

$$F_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m (\mathbf{x}_k - \mathbf{v}_i^{(l)})(\mathbf{x}_k - \mathbf{v}_i^{(l)})^T}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \forall 1 \leq i \leq c \tag{3.34}$$

Add a scaled identity matrix by adjusting the value of γ in equation 3.35. The reason for this addition is for cases where the clusters are extremely extended in the direction of the largest eigenvalues. The computed covariance matrix cannot estimate the underlying data distribution.

$$F_i := (1 - \gamma) F_i + \gamma (F_0)^{\frac{1}{n}} I \tag{3.35}$$

Extract the eigenvalues (λ_{ij}) and the eigenvectors (ω_{ij}) of F and find the maximum eigenvalue $(\lambda_{i,max})$. Set $\lambda_{i,max} = \frac{\lambda_{ij}}{\beta}, \forall j$ for which $\frac{\lambda_{i,max}}{\lambda_{ij}} \geq \beta$. The introduction of β is due to condition number problems with respect to the covariance matrix (in some cases it becomes nearly singular). For this reason, the maximum and minimim eigenvalue ratio should be constrained to a predefined threshold, i.e. β. Reconstruct F_i by making use of equation 3.36.

$$F_i = \Omega \Lambda \Omega^{-1} \tag{3.36}$$

where Λ is a diagonal matrix containing all the eigenvalues and Ω contains all the eigenvectors.

3. Compute the distances by using equation 3.37.

$$D_{ikB_i}^2(\mathbf{x}_k, \mathbf{v}_i) = \left(\mathbf{x}_k - \mathbf{v}_i^{(l)}\right)^T \left[(\rho_i \det(F_i))^{\frac{1}{n}} F_i^{-1}\right] \left(\mathbf{x}_k - \mathbf{v}_i^{(l)}\right) \tag{3.37}$$

4. Update the partition matrix with equation 3.38.

$$\frac{1}{\mu_{ik}^{(l)}} = \sum_{j=1}^{c} \left(\frac{D_{ikB_i}(\mathbf{x}_k, \mathbf{v}_i)}{D_{jkB_i}(\mathbf{x}_k, \mathbf{v}_j)}\right)^{\frac{2}{(m-1)}}, \forall 1 \leq i \leq c, 1 \leq k \leq N \tag{3.38}$$

5. Repeat steps (1) - (4) until $\|U^{(l)} - U^{(l-1)}\| < \epsilon$.

Figure 3.4 shows the clusters found by this algorithm on a synthetic overlapping data set. When figure 3.4 is compared to figure 3.3, it is seen that the GK algorithm performs slightly better with clustering the elongated data at the bottom of the figure. As mentioned earlier, this is due to the fact that the algorithm makes provision for different shapes with respect to the clusters.

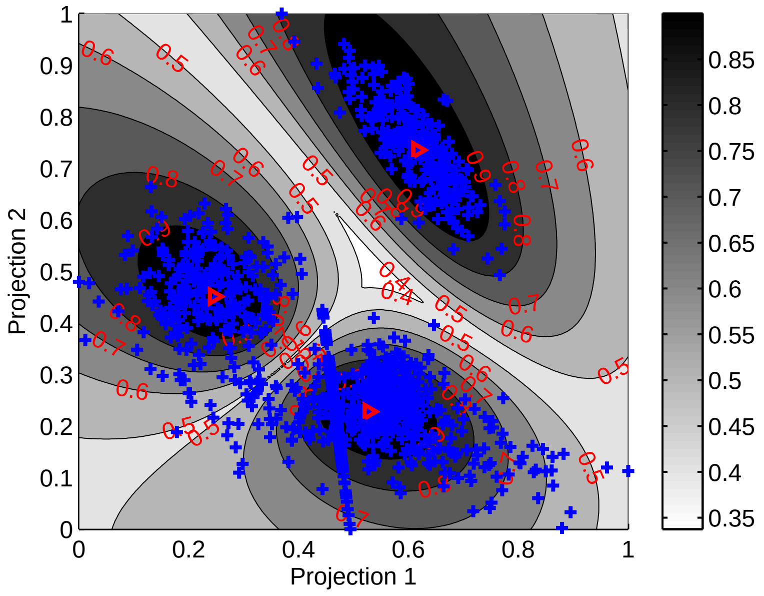


Figure 3.4: Result of the Gustafson-Kessel algorithm by using a synthetic overlapping data set. The gradient line presented on the figure are linear representations of the partitioning between different clusters.

3.2.4 Gath-Geva clustering algorithm

The fuzzy maximum likelihood estimates (FMLE) clustering algorithm makes use of an FMLE distance norm (Balasko et al., 2005). This norm is shown in equation 3.39.

$$D_{ik}(\mathbf{x}_k, \mathbf{v}_i) = \frac{\sqrt{\det(F_i)}}{\alpha_i} \exp\left(\frac{1}{2}(\mathbf{x}_k - \mathbf{v}_i)^T F_i^{-1}(\mathbf{x}_k - \mathbf{v}_i)\right) \tag{3.39}$$

This distance norm includes an exponential term, which means that the distance norm decreases faster than the inner-product norm found in the Gustafson-Kessel algorithm.

Derivation

Let F_i denote the fuzzy covariance matrix of the i-th cluster. This is given by equation 3.40.

$$F_i = \frac{\sum_{k=1}^{N} (\mu_{ik})^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^T}{\sum_{k=1}^{N} (\mu_{ik})^m}, \forall 1 \leq i \leq c \tag{3.40}$$

Balasko et al. (2005) notes that the normal weighting for the FMLE algorithm is $m = 1$. In cases where $m = 2$ is used, the partition becomes more fuzzy to compensate for the exponential term of the distance norm. The difference between the Gustafson-Kessel algorithm's F_i and that of Gath-Geva is that the latter can be weighted.

α_i is a factor related to the prior probability of selecting cluster i. This is given by equation 3.41.

$$\alpha_i = \frac{1}{N} \sum_{k=1}^{N} \mu_{ik} \tag{3.41}$$

The membership degrees, μ_{ik}, are interpreted as the posterior probabilities of selecting the i-th cluster given a data point $\mathbf{x}_i$. This algorithm is able to detect clusters of varying sizes, densities and shapes (Balasko et al., 2005) but unfortunately is less robust in the sense that it needs good initialization. This is due to the exponential term within the distance norm which leads to convergence near a local minimum.

Implementation of algorithm

Given a data set, X, specify c and choose a weighting exponent $m > 1$ as well as a termination tolerance ϵ. Initialise the partition matrix.

Repeat for $l = 1, 2, \ldots$

1. Compute the cluster centers by means of equation 3.42.

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m \mathbf{x}_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \forall 1 \leq i \leq c \tag{3.42}$$

2. Compute the cluster covariance matrices by using equation 3.43. The distance to the prototype is calculated based on the fuzzy covariance matrices of the cluster.

$$F_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m (\mathbf{x}_k - \mathbf{v}_i^{(l)})(\mathbf{x}_k - \mathbf{v}_i^{(l)})^T}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \forall 1 \leq i \leq c \tag{3.43}$$

The distance function is shown in equation 3.44 (Balasko et al., 2005).

$$D_{ik}(\mathbf{x}_k, \mathbf{v}_i) = \frac{\sqrt{\det(F_i)}}{\alpha_i} \exp\left(\frac{1}{2}(\mathbf{x}_k - \mathbf{v}_i^{(l)})^T F_i^{-1}(\mathbf{x}_k - \mathbf{v}_i^{(l)})\right) \tag{3.44}$$

with α_i given by equation 3.41.

3. Update the partition matrix with equation 3.45.

$$\frac{1}{\mu_{ik}^{(l)}} = \sum_{j=1}^{c} \left(\frac{D_{ik}(\mathbf{x}_k, \mathbf{v}_i)}{D_{jik}(\mathbf{x}_k, \mathbf{v}_j)}\right)^{\frac{2}{(m-1)}}, \forall 1 \leq i \leq c, 1 \leq k \leq N \tag{3.45}$$

4. Steps (1) - (3) are repeated until $\|U^{(l)} - U^{(l-1)}\| < \epsilon$.

Figure 3.5 shows the output of this algorithm.

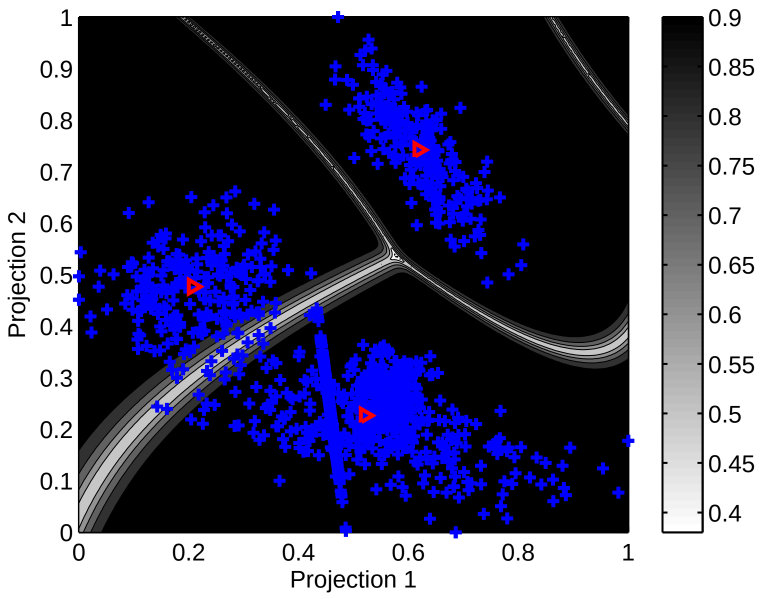


Figure 3.5: Result of the Gath-Geva algorithm by using a synthetic overlapping data set. The gradient line presented on the figure are linear representations of the partitioning between different clusters.

3.3 Cluster Validation

Cluster validation refers to the question of how well a particular clustering algorithm clusters the particular data. Given a data set and the associated parameter set of the algorithm, it will always attempt to find the optimal geometric cluster fit of the data. This does not guarantee that the best fit would result in something meaningful. Problems with clustering may be due to an inefficient number of clusters or to cluster shapes not corresponding the underlying data structure (Wang & Zhang, 2007).

Either the number of clusters might not be in the correct quantity or the cluster shapes might not correspond to the underlying group structure of the data. Therefore, we define the following two concepts (Wang & Zhang, 2007; Bensaid et al., 1996):

Compactness This defines the closeness of the cluster elements. A typical implementation is the variance between the elements. The variance gives an indication of how different the members are. Therefore, a smaller variance indicates closer cluster elements.

Separation This gives an indication of how distinct clusters are. It makes use of the distance between clusters. This measure's computational efficiency has led to its wide application.

Two of the most common courses of action to determine the appropriate number of clusters for the given data set are listed below.

- Starting of with a sufficiently large number of clusters, successively reduce the number by merging clusters that are similar. The "goodness" of fit is determined with some predefined criteria. This is also known as *compatible cluster merging.*

- Cluster the data set with varying values for c. Assess the "goodness" of fit by making use of different *validity measures*. Balasko et al. (2005) provides two options for this process.

  The first approach defines a validity function which evaluates the complete partition. An upper limit to the number of clusters to be tested for should be selected (c_{max}). The desired clustering algorithm is then run with $c \in \{2, 3, \ldots, c_{max}\}$. This results in a validity measure for each number of c which can be individually compared to the others.

  The second approach relies on the definition of a validity function that evaluates individual clusters of a cluster partition. As with the first approach, c_{max} needs to be estimated and the cluster analysis has to be carried out for c_{max}. The resulting clusters are compared to each other on the basis of a validity function. Similar clusters are collected in one cluster and very bad clusters are eliminated. This reduces the number of clusters and can be repeated until only well defined clusters remain.

Many scalar validity measures exist. Wang & Zhang (2007) found that no one validation index could accurately determine the optimum number of clusters with all the data sets they used. Pal & Bezdek (1995) stated, "no matter how good your index is, there is a data set out there waiting to trick it (and you)". For this reason, more than one will be used as validity functions in every clustering.

The Partition Coefficient (PC) and Classification Entropy (CE) (also known as Partition Entropy) are some of the first validity indices (χ) to be associated with fuzzy clustering algorithms.

$$\chi_{PC}(U) = \frac{1}{N}\sum_{i=1}^{c}\sum_{j=1}^{N}(\mu_{ij})^2 \tag{3.46}$$

$$\chi_{CE}(U) = \frac{1}{N}\sum_{i=1}^{c}\sum_{j=1}^{N}\mu_{ij}\log\mu_{ij} \tag{3.47}$$

These indices measure the fuzziness of the cluster partition or the overall average overlap between pairs of fuzzy subsets (Xie & Beni, 1991). Pal & Bezdek (1995) showed that in the case of a hard partition clustering algorithm, these values would take their maximum and minimum respectively at $U = 1/c \doteq \overline{U}$ of M_{fc}. $\overline{U}$ is the fuzziest partition since it assigns every point in X, to all c classes with equal membership values $1/c$. When the clustering algorithm produces a partition U that is close to $\overline{U}$, it is not finding a well defined substructure in X. This could be because the clustering algorithm is at fault, or there's no well defined substructure within X. Consequently, the minimum in χ_{PC} and maximum in χ_{CE} is helpful in deciding when no structure is found, but yields no clear indication of the substructure when U approaches M_{hc}. Since $\chi_{PC} = 1$ and $\chi_{CE} = 0$ for every U in M_{hc}, that is in a hard partition, it is incorrect to assume that when $\chi_{PC} = 1$ and $\chi_{CE} = 0$, U is a good clustering of X (Pal & Bezdek, 1995). Xie & Beni (1991) warn that these validity measures lack a direct connection to the data. The Partition Index (SC) is the sum of the ratio between the compactness and separation of the clusters. The Separation Index (S), also known as the Xie and Beni index, differs from the SC only in the denominator.

$$\chi_{SC}(U,V,X) = \frac{\displaystyle\sum_{i=1}^{c}\sum_{j=1}^{N}(\mu_{ij})^m\|\mathbf{x}_j - \mathbf{v}_i\|^2}{\displaystyle N_i\sum_{k=1}^{c}\|\mathbf{v}_k - \mathbf{v}_i\|} \tag{3.48}$$

$$\chi_S(U,V,X) = \frac{\displaystyle\sum_{i=1}^{c}\sum_{j=1}^{N}(\mu_{ij})^2\|\mathbf{x}_j - \mathbf{v}_i\|^2}{\displaystyle N\min_{i\neq k}\|\mathbf{v}_k - \mathbf{v}_i\|} = \left[\frac{\frac{\sigma}{N}}{\mathrm{sep}(V)}\right] \tag{3.49}$$

Bensaid et al. (1996) notes two important differences when comparing the SC and the S.

1. A good (U,V) pair should produce a small value of σ because μ_{ij} is expected to be large when $\|\mathbf{x}_j - \mathbf{v}_i\|$ is low. Well separated $\mathbf{v}_i$s will result in a high value of $\mathrm{sep}(V)$. Therefore, when $\chi_S(U_1,V_1,X) < \chi_S(U_2,V_2,X)$, it is considered to be a better partition of the data Pal & Bezdek (1995). Experiments showed that this feature may be useful when searching for the right number of clusters. It does not

seem to be as useful when comparing different partitions having an equal number of clusters. This criterion favours the creation of a set of clusters that are maximally separate from each other. This will minimise the fuzziness in assignments that are close to each other.

2. SC is the sum of the individual cluster validity measures, normalised through division by the fuzzy cardinality, $N_i = \sum_{k=1}^{c} \mu_{ik}$, of each cluster. This normalisation is aimed at making the Partition Index more insensitive to variations in cluster sizes, a desirable property not shared by the clustering objective function, J. This validity measure is designed to complement the objective function.

Dunn's Index (DI) is based on geometrical considerations in that it is designed to identify sets of clusters that are compact and well separated. Bezdek & Pal (1995) defines it as follows: Let S and T be non-empty subsets in $\mathbb{R}^n$, and let $\mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}^+$ be any metric. The standard definitions of the diameter, Δ, of S and the set distance, δ between S and T are shown in equations 3.50 and 3.51.

$$\Delta(c) = \max_{x,y \in S}\{d(x,y)\} \tag{3.50}$$

$$\delta(S,T) = \min_{x \in S, y \in T}\{d(x,y)\} \tag{3.51}$$

For any partition $U \leftrightarrow X = X_1 \cup \ldots X_i \cup \ldots X_c$, Dunn defined the separation index of U as shown in equation 3.52.

$$\chi_{DI}(U,X) = \min_{i \in c}\left\{ \min_{j \in c, i \neq j}\left\{ \frac{\delta(X_i, X_j)}{\max\limits_{k \in c}\{\Delta(X_k)\}} \right\}\right\} \tag{3.52}$$

Here, $\delta(X_i, X_j)$ measures the distance between clusters directly on the points in the clusters and $\Delta(X_k)$ measures the scatter volume for cluster X_k. Large values in χ_{DI} indicate good clusters.

To make the calculation of Δ simpler, Balasko et al. (2005) propose that is be replaced by $\delta_G = \max\limits_{x,y \in S} |d(y, v_j) - d(x, v_j)|$, where v_j is the cluster center of the j-th cluster. This is known as the alternative Dunn Index (ADI)

$$\chi_{ADI} = \min_{i \in c}\left\{ \min_{j \in c, i \neq j}\left\{ \frac{\delta_A(X_i, X_j)}{\max\limits_{k \in c}\{\Delta(X_k)\}} \right\}\right\} \tag{3.53}$$

Singhal & Seborg (2005) suggest a completely different approach to determine the optimum number of clusters. It should be noted that they clustered different data sets and not one, continuous data set. The number of clusters, c, is increased to the total number of data sets, Q.

For each iteration, they calculate a pre-defined dissimilarity [2]. The resulting dissimilarity sequence, $J(c)$ is then used to estimate the optimal number of clusters. They note that it is typical to find that J decreases with an increase in c. However, the optimum number of clusters can be found when $J(c)$ changes significantly. With this in mind, they defined the following property,

$$\psi(c) = \text{sign}[\text{d}J(c+1) - \text{d}J(c)], c = 1, 2, 3, \ldots \tag{3.54}$$

where

$$\text{d}J(c) = \frac{|J(c+1) - J(c)|}{J(c)} \times 100\%, c = 1, 2, 3, \ldots \tag{3.55}$$

The value for which $\text{d}J(c)$ reaches a minimum or is close to zero, is referred to as the 'knee' of the sequence. The values where the plot of $J(c)$ has a knee are possible values for the optimum number of clusters. For this reason, the sign of the difference between successive values of $\text{d}J(c)$ is used to estimate the locations of the 'knees'. The values for c which changes the sign of $\psi(c)$ from negative to positive are considered as optimum values. Singhal & Seborg (2005) suggests that the first knee is usually selected as the optimum.

For the purposes of this project, their dissimilarity index was not used as they were clustering data sets instead of observations. For this reason, it is replaced with the normal euclidean dissimilarity or distance.

3.4 Dimensional Reduction

The topics discussed in this section can also be referred to as feature extraction techniques. Dimensional reduction is an important task in machine learning as it facilitates classification, compression and visualisation of high dimensional data by mitigating undesired properties of high dimensional spaces. These techniques consists of mapping input vectors (of observations $[\mathbf{x} \in \mathbb{R}^n]$, e.g. PV's, CV's) onto a new feature space which is more suitable for a given task ($\mathbf{y} \in \mathbb{R}^o$).

3.4.1 Principal component analysis

Principal component analysis (PCA) is a tool which reduces the dimensionality of the problem by defining a series of new axes called principal components, along the directions of the maximum variation in the data. It should be noted that this is a linear process, therefore the new vectors are a linear combination of the original variables. These vectors are also orthogonal to each other (Martin & Morris, 1996). Franc & Hlavac (2004) also note

[2]Singhal & Seborg (2005) defined their dissimilarity function in terms of a PCA calculated using the k largest principal components

that PCA is an unsupervised learning algorithm. Equation 3.56 shows the decomposition according to Martin & Morris (1996)

$$X = TP^T = \sum_{i=1}^{m} \mathbf{t}_i \mathbf{p}_i^T \tag{3.56}$$

The results of a PCA analysis can be broken up into the following:

- The first principal component is that which describes the most variability.
- The loadings ($\mathbf{p}_i$) defines the directions of most variability.
- The score vector ($\mathbf{t}_i$) provides the relationship of the projection of each principal onto $\mathbf{p}_i$.
- The second principal component is orthogonal to the first and represents the second most variability in the data. This process continues until n principal components are obtained.

PCA has found wide application in industry, including facial recognition, coin classification and seismic series analysis (van der Maaten et al., 2007).

3.4.2 Sammon Mapping

Both PCA and Sammon Mappings are global techniques for dimensionality reduction. These techniques attempt to preserve global properties of the data. The main difference between PCA and Sammon Mapping is that it is capable of constructing nonlinear transformations between the high dimensional and low dimensional spaces. Phillpotts (2007) found that kernel-PCA techniques did not perform as well as linear PCA.

Sammon mapping is a nonlinear mapping algorithm which has been found highly effective in the analysis of multivariate data. This mapping technique is based upon a point mapping of the N n-dimensional vectors from the n-space to a lower dimensional space such that the inherent structure of the data is approximately preserved under the mapping (Sammon, 1969).

The Sammon algorithm is well known for its ability to find good lower dimensional representations of X. There are however a few limiting factors to this algorithm.

- The algorithm solves $N \times n$ simultaneous, coupled nonlinear equations making it computationally intensive (Bezdek & Pal, 1995).
- By making use of the gradient based procedure to search for the minimum Sammon stress (or Sammon error function), a local minimum on the error surface could be reached Lerner et al. (1998). Therefore, a significant number of experiments with

different random initialisations may be required. Methods that attempt to solve this problem include basing the initialisation on information obtained from the data such as the first and second norms of the feature vectors or the principal axes of the covariance matrix of the data.

- When a new point has to be mapped, the whole mapping procedure has to be repeated. This would make it impractical for real-time data mining (Feil et al., 2007).

Derivation

Suppose we have N vectors in an n-space designated $\mathbf{x}_i, i = 1, 2, 3...N$. We also define N vectors in a o-space designated $Y_i, i = 1, 2, 3...N$. Let the distance between the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ be defined by $d^*_{ij} = \text{dist}[\mathbf{x}_i, \mathbf{x}_j]$ and the corresponding distance in the lower dimensional space be defined as $d_{ij} = \text{dist}[\mathbf{y}_i, \mathbf{y}_j]$. At this point it is worth mentioning that any distance metric could be used. However, if no before hand knowledge of the data is known, there is no reason not to use the Euclidean distance metric.

The o-space configuration is calculated by projecting the L-dimensional data orthogonally onto the o-space spanned by o original coordinates with the largest variation. This configuration is shown in equation 3.57.

$$\mathbf{y}_1 = \begin{bmatrix} y_{11} \\ \vdots \\ y_{1d} \end{bmatrix} \mathbf{y}_2 = \begin{bmatrix} y_{21} \\ \vdots \\ y_{2d} \end{bmatrix} \mathbf{y}_3 = \begin{bmatrix} y_{N1} \\ \vdots \\ y_{Nd} \end{bmatrix} \tag{3.57}$$

When this is complete, all the o-space inter-point distances are calculated which is used to calculate an error, E. This represents how well the N points in the o-space fits N points in the n-space. This is shown in equation 3.58.

$$E = \frac{1}{\sum\limits_{i<j} d^*_{ij}} \sum_{i<j}^{N} \frac{|d^*_{ij} - d_{ij}|^2}{d^*_{ij}} \tag{3.58}$$

The final step is to change Y, or equivalently, the o-space configuration to minimise the error. The minimisation of E is an optimisation problem in $o \times N$ variables $y_{pq}, p = 1, \ldots, N$ and $q = 1, \ldots, d$. The steepest descent[3] method is used to search for a minimum error (Sammon, 1969). Equation 3.59 shows the minimisation problem at the t-th iteration.

$$y_{pq}(t+1) = y_{pq}(t) - \theta \left[\frac{\frac{\partial E(t)}{\partial y_{pq}(t)}}{\frac{\partial^2 E(t)}{\partial y_{pq}(t)^2}} \right] \tag{3.59}$$

[3]This algorithm performs a line search in the direction of the steepest decent of the function at the current location. This is done at each iteration.

where the partial derivatives are shown in equations 3.60 and 3.61.

$$\frac{\partial E(t)}{\partial y_{pq}(t)} = -\frac{2}{m} \sum_{j=1, j\neq p}^{N} \left[\frac{d^*_{pj} - d_{pj}}{d_{pj} d^*_{pj}} \right] (y_{pq} - y_{jq}) \tag{3.60}$$

$$\frac{\partial^2 E(t)}{\partial y_{pq}(t)^2} = -\frac{2}{w} \sum_{j=1, j\neq p}^{N} \frac{1}{d_{pj} d^*_{pj}} \left[(d^*_{pj} - d_{pj}) - \frac{(y_{pq} - y_{jq})^2}{d_{pj}} \left(1 + \frac{d^*_{pj} - d_{pj}}{d_{pj}} \right) \right] \tag{3.61}$$

where w is the constant $\sum_{i<j}^{N} d^*_{ij}$ in equation 3.58.

3.4.3 Fuzzy Sammon Mapping

To overcome the problem of re-mapping a complete data set every time a new data point is added to the data set, Feil et al. (2007) tailored the original Sammon mapping for visualisation of fuzzy clustering results.

When using fuzzy clustering algorithms, only the distance between the data points and the cluster centers are considered to be important. Unlike the Sammon mapping, the fuzzy Sammon mapping maps the clusters and data points such that the distances between the cluster centers and the data is preserved (Yu et al., 2006). This is weighted by the fuzzy membership function and the modified error function is shown in equation 3.62.

$$E_{\text{fuzz}} = \sum_{i=1}^{c} \sum_{k=1}^{N} (\mu_{ki})^m (d(\mathbf{x}_k, \eta_i) - d^*_{ki})^2 \tag{3.62}$$

where $d(\mathbf{x}_k, \eta_i)$ represents the distance between the $\mathbf{x}_k$ data point and the η_i cluster center in the original dimensional space, while $d^*_{ki} = d^*(\mathbf{y}_k, \mathbf{z}_i)$ represents the Euclidean distance between the projected cluster center $\mathbf{z}_i$ and the projected data $\mathbf{y}_k$. The consequence of this is that in the projected space, every cluster is represented by a single point, regardless of the form of the original cluster prototype (Feil et al., 2007).

The computational procedure for the fuzzy Sammon mapping is shown below (Feil et al., 2007) (Yu et al., 2006):

1. Initialise the projected data points by $\mathbf{y}_k$ by PCA based projection of $\mathbf{x}_k$, and compute the projected cluster centers by using equation 3.23. After this, compute the distances, D^*, with the use of these projected points.

2. Compute the partial derivatives shown in equations 3.60 and 3.61 and update y_{pq} in equation 3.59.

3. Recalculate the projected cluster centers, $\mathbf{z}_i$ and update the projected distance matrix, D^*.

4. Compute E_{fuzz} with equation 3.62 while using equations 3.60 and 3.61 as the derivatives.

5. Continue until either the maximum number of iterations are reached or when $E_{\text{fuzz}} < \epsilon$.

Feil et al. (2007) claim that resultant two dimensional plot is easily interpretable since it is based on a normal Euclidean distance measure between the cluster centers and the data points.

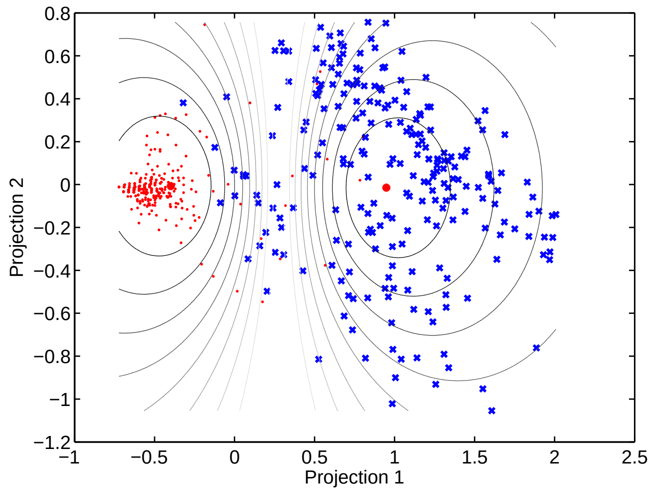
3.4.4 Dimensional reduction metrics

Of course the resulting two dimensional plot will only approximate the original higher dimensional data. To get a qualitative estimation of how well the dimensional reduction algorithms projected the data to the lower dimensional space, the mean square error of the original and the recalculated membership functions can be calculated (Feil et al., 2007). This is shown in equation 3.63.

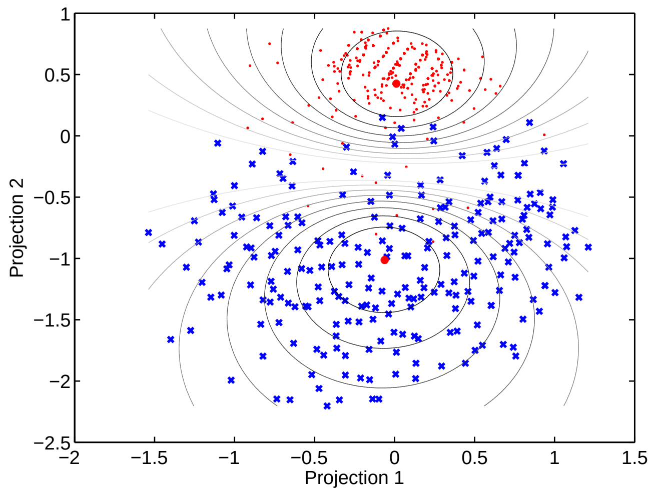
$$p = \|U - U^*\| \tag{3.63}$$

There are many other tools to qualify the quality of the clusters mappings. The validity measures defined in section 3.3 could be used to calculate the cluster validity before and after the mappings.

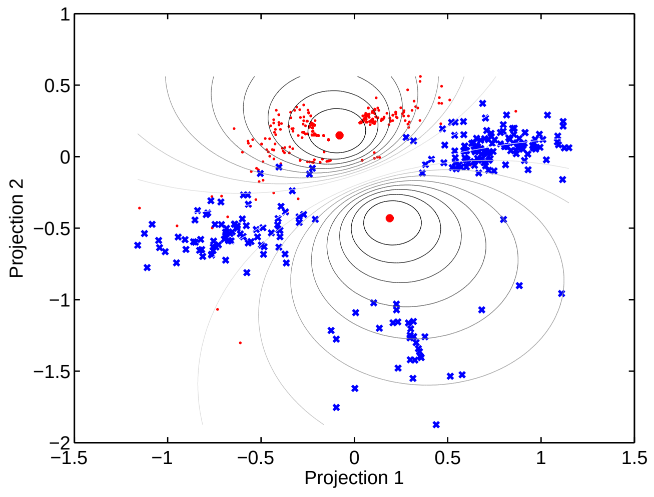
Figure 3.6 illustrates the dimensional reduction techniques discussed above. The Wisconsin Breast Cancer data set was used together with two cluster centers. The performance metric, p, of each dimensional reduction technique is also shown. From this, the fuzzy Sammon mapping performs the best, followed by the original Sammon mapping and lastly the principal component analysis.



(a) PCA, $p = 0.1422$



(b) Sammon, $p = 0.1182$



(c) Fuzzy Sammon, $p = 0.0163$

Figure 3.6: Examples of dimensional reduction techniques. The Wisconsin Breast Cancer data set was used together with the fuzzy C-means clustering algorithm with $c = 2$. The associated performance metric is also given.

CHAPTER 4

Time Delay Estimation

We define the concept of Multiple Signal Time Delay Estimation (MSTDE). We develop this algorithm from existing cross-correlation and correlation techniques and combine it with recently published statistical thresholding methods to determine the significance of signal correlations. We then show how linear algebra can be used to solve for a single set of time shifts by either minimising the norm of the time shifts (for a rank deficient system) or the norm of the residual (excess rank).

Chemical processes can be large and complex systems with many measurements and control actions. A process fault or disturbance could appear in many places on a plant. These faults and disturbances propagate through the plant due to a causal link. A sudden loss of the steam pressure in a distillation column's boiler will affect several variables including the plate temperatures, the liquid level in the condenser drum and the top composition. Although there is only one root cause, the effects are visible in many of the measurements made (Bauer & Thornhill, 2008).

If there is a strong causal relationship between these variables, it is intuitive to assume that these events will be detected at certain times after the initial event has occurred. These times are known as the process dead times or lags and are illustrated in figure 4.1

Problem Statement: Determine whether and by how much a set of signals should be shifted to aid a clustering algorithm, given the causal links between the signals.

4.1 Time delay estimation

Time Delay Estimation (TDE) refers to the calculation of a time delay between a pair of signals and its detection poses a challenge. TDE has many applications including sonar and radar (Knapp & Carter, 1976), in wide-band wireless communication systems (Liu

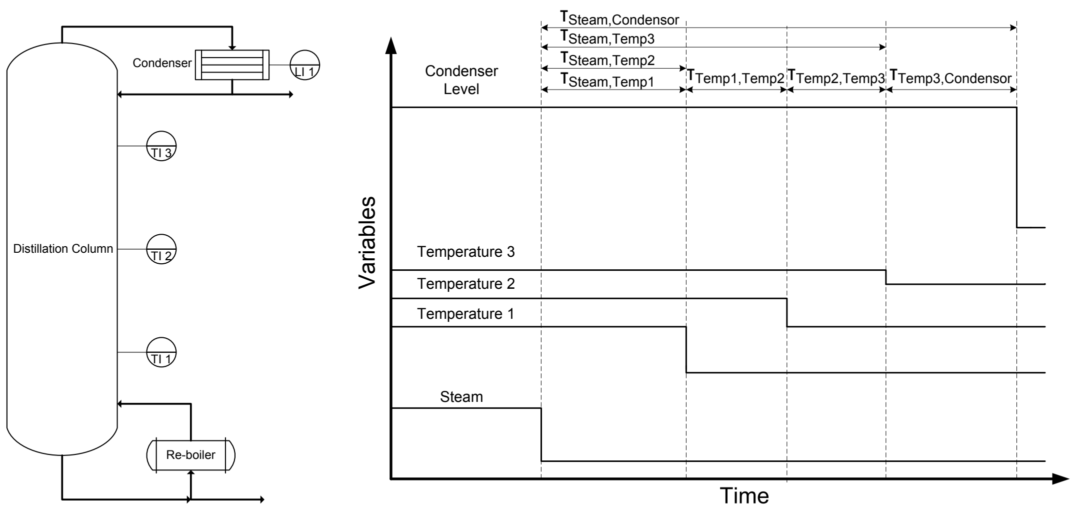


Figure 4.1: Example of casual process measurements and their associated time delays. Note that not all the inter-signal time delays are shown.

et al., 2008), sensor networks (Ash & Moses, 2005) and many more. Knapp & Carter (1976) pose the problem as follows: A signal emanating from a remote source (with its associated noise) and monitored at different spatial positions can be modelled as shown in equation 4.1,

$$\begin{aligned} x_1(t) &= s_1(t) + n_1(t) \\ x_2(t) &= \alpha s_1(t+D) + n_2(t) \end{aligned} \tag{4.1}$$

where $s_1(t)$, $n_1(t)$, and $n_2(t)$ are real, jointly stationary random processes. It is important to note that $s_1(t)$ is assumed to be uncorrelated with the noise present in the system. This system differs from a typical chemical process. In a chemical processes, various state variables (i.e. pressure, temperature etc.) are measured and deviations in some of these variables could be due to one or more independent sources.

4.1.1 Cross Correlation Function (CCF)

Knapp & Carter (1976) showed that for a certain choice of the weighting function, the cross-correlation function method of determining signal delays is equivalent to that of more complex processors including the maximum likelihood estimator, Eckart filter etc. The advantages of using the cross-correlation function is:

- Ease of implementation.
- Familiarity of the function among practising engineers (Bauer & Thornhill, 2008).
- Noise tolerance (Bauer, 2005).

The cross correlation function between two, wide-sense stationary signals is shown in equation 4.2 (Bauer & Thornhill, 2008; Barkat, 1991: p.74-76).

$$\phi_{xy}[\kappa] = \frac{1}{N-\kappa} \sum_{i=-N}^{+N} \hat{\mathbf{x}}_i \hat{\mathbf{y}}_{i+\kappa} \tag{4.2}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ are mean centred and variance scaled derivatives of the original signals $\mathbf{x}$ and $\mathbf{y}$ and κ is the CCF time delay. The cross correlation function is not symmetrical or commutative (Bauer, 2005).

To calculate the inter-signal time delays, we follow the following procedure for each pair of signals as described by Bauer & Thornhill (2008) and Bauer (2005).

1. Record both the minimum and maximum values obtained from the cross-correlation function.

$$\begin{aligned} \phi^{max} &= \max_{\kappa}\{\phi_{xy}[\kappa]\}, \\ \phi^{min} &= \min_{\kappa}\{\phi_{xy}[\kappa]\}, \end{aligned} \tag{4.3}$$

   where ϕ^{max} and ϕ^{min} are positive and negative respectively. The corresponding time delays (κ^{max} and κ^{min}) are also recorded. Bauer (2005) states that ϕ^{max} will always be positive and ϕ^{min} will always be negative.

2. Assign the actual time delay for the particular signals. The choice depends on the absolute values of ϕ^{max} and ϕ^{min}.

$$\tau_{ij} = \begin{cases} \kappa^{max}, & \phi^{max} + \phi^{min} \geq 0, \\ \kappa^{min}, & \phi^{max} + \phi^{min} < 0. \end{cases} \tag{4.4}$$

The result of this is a time delay matrix (Δ) which yields the same information as figure 4.1 and contains a comprehensive set of delays (τ) between all the relevant signals. This is shown in equation 4.5.

$$\Delta = \begin{array}{cccccc} S & T_3 & T_2 & T_3 & C & \\ 0 & \tau_{ST_1} & \tau_{ST_2} & \tau_{ST_3} & \tau_{SC} & S \\ & 0 & \tau_{T_1T_2} & \tau_{T_1T_3} & \tau_{T_1C} & T_1 \\ & & 0 & \tau_{T_2T_3} & \tau_{T_2C} & T_2 \\ & & & 0 & \tau_{T_3C} & T_3 \\ & & & & 0 & C \end{array} \tag{4.5}$$

4.1.2 Statistical significance and weighting

Bauer & Thornhill (2008) developed empirical formulae to test the significance of correlation values and signal directionality. These formulae are based on hypothesis testing as well as the number of samples in a time series. To determine the statistical significance for a pair of signals, we use the following procedure:

The cross-correlation function of a finite signal has a maximum and minimum value even when no time delay exists. For this reason it is necessary to establish whether there is a statistical significance between x_i and $y_{i+\tau}$. The maximum time delayed correlation between two time series is

$$\rho = \max\{\phi^{max}, |\phi^{min}|\}. \tag{4.6}$$

This is tested against a threshold value, ρ_{th}. The correlation threshold is

$$\rho_{th}(N) = 1.85N^{-0.41} + 2.37N^{-0.53}. \tag{4.7}$$

where N is the number of samples in the given data set.

The CCF of a time delayed harmonic oscillation is periodic with the same frequency as the oscillation. Certain ambiguities arise form this behaviour due to phase warping. For this reason, no estimate of the time delay should be offered. For harmonic time series, ϕ^{max} and ϕ^{min} of the CCF are of equal magnitude. Therefore, one failure of the directionality test is the presence of an oscillation. Directionality can only be inferred in a time delayed oscillation if some additional dynamic features are present (Bauer & Thornhill, 2008). In order to confirm directionality, the magnitudes of ϕ^{max} and ϕ^{min} have to differ significantly. A directionality index is introduced.

$$\psi = 2\frac{|\phi^{max} + \phi^{min}|}{\phi^{max} + |\phi^{min}|} \tag{4.8}$$

If the index is small, then no decision can be made because ϕ^{max} and ϕ^{min} have the same magnitude. In a similar fashion to the correlation threshold, a directionality threshold value is calculated and ψ is tested against ψ_{th}. The directionality threshold is

$$\psi_{th}(N) = 0.46N^{-0.16}. \tag{4.9}$$

If $\psi \geq \psi_{th}$ and $\rho \geq \rho_{th}$, then the delay is statistically significant.

4.1.3 Optimal signal time delay

To determine a single signal shift from Δ, we propose the following conjecture.

Conjecture 1 (Optimal signal shift) *If Δ_{ij} gives an accurate estimate of the inter-*

signal time delays, and if the time delays are sufficiently described by M input or output lags, where M is the number of signals, then we can write

$$l_j - l_i = \Delta_{ij}, \forall i = 1..M, j > i.$$

Noticing that this is a linear system, the problem can be defined as follows:

$$Al = L^*$$

In a large data set there are uncorrelated signals. If the linear set of equations are solved, each signal, irrespective of its contribution towards the causality, is weighted equally. A weighting function assigns weights to the individual equations (with respect to their contribution towards the causality as determined from the CCF in this case) to ensure prominence to more important signals in the algorithm. This could also reduce the number of equations that need to be solved, ensuring an even better efficiency for the algorithm.

Using the maximum correlation threshold and the directionality threshold, together with the actual correlation coefficients, we propose the following weighting function for the multiple signal time delay estimation (MSTDE) algorithm:

$$w_i = \begin{cases} [\rho_{ij}, j = i+1 \ldots N, i = 1 \ldots N], & \psi \geq \psi_{th} \text{ and } \rho \geq \rho_{th}, \\ 0, & \psi < \psi_{th} \text{ and } \rho < \rho_{th} \end{cases} \tag{4.10}$$

where ρ_{ij} is the correlation between signals i and j. The weight function becomes,

$$W = \begin{pmatrix} w_1 & 0 & \ldots & 0 \\ 0 & w_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & w_N \end{pmatrix} \tag{4.11}$$

The problem is now defined as shown in equation 4.12.

$$WAl = WL^* \tag{4.12}$$

In general, WA is not square, as there are $\frac{M(M-1)}{2}$ permutations of the M variables that have to be considered. Also, A is not invertible, as there is not enough independent equations to guarantee a unique solution. To compensate for this, we add one equation, $l_1 = 0$ and solve in a least-square sense. Therefore, the pseudo-inverse is used to

- minimise $|l|$ for a rank deficient system or
- minimise the residual, $|W(Al - L^*)|$, for rank excess.

The solution is tested by calculating the resulting residual norm. This would ideally be as close to zero as possible. After calculating the shifts with $l_1 = 0$, we define $\hat{l} = l - \min_i l_i$ to obtain strictly positive shifts. Algorithm 1 shows implementation of the MSTDE algorithm.

Algorithm 1 MSTDE Algorithm

```
Require: $N$ signals, $x_i$, $\Delta_{ij} = 0, i \leq j$
 1: for $i = 1 \ldots N-1$ do
 2:     for $j = i+1 \ldots N-1$ do
 3:         Calculate $\phi^{max}$, $\phi^{min}$, $\kappa^{min}$ and $\kappa^{max}$      ▷ equation 4.3
 4:         Calculate $\Delta_{ij}$      ▷ equation 4.4
 5:         Calculate $\rho$ and $\rho_{th}$      ▷ equations 4.6 and 4.7
 6:         Calculate $\psi$ and $\psi_{th}$      ▷ equations 4.8 and 4.9
 7:         $r = (i-1)N + j - i$
 8:         $A_{r,i} = -1$
 9:         $A_{r,j} = 1$
10:         $L^*_r = \Delta_{ij}$
11:         $\mathbf{w}_r = \begin{cases} \rho_{ij}, & \psi \geq \psi_{th} \text{ and } \rho \geq \rho_{th}, \\ 0, & \psi < \psi_{th} \text{ and } \rho < \rho_{th} \end{cases}$
12:     end for
13: end for
14: $\text{diag}(W) = \mathbf{w}$
15: $l = A^{\dagger} W^{\dagger} W L^*$      ▷ $A^{\dagger}$ and $W^{\dagger}$ constitute the Moore-Penrose pseudo-inverse
16: $\hat{l} = l - \min_i(l_i)$
```

Illustrative Example

Consider the simplified system shown in figure 4.2.

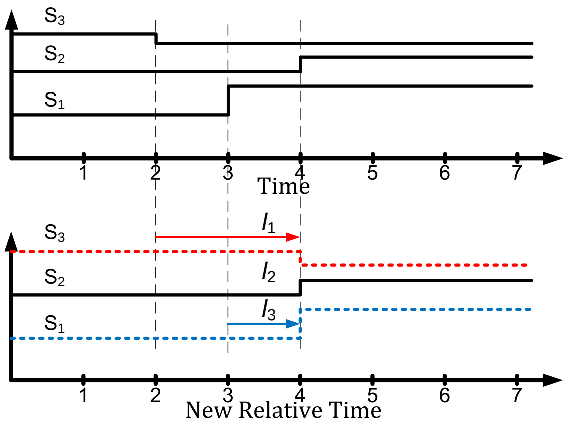


Figure 4.2: Illustrative example of the optimal time shift for an arbitrary set of signals

Here follows the individual steps to calculate the optimal time shift.

1. From figure 4.2, we can easily construct the time delay matrix assuming all delays are significant. This is shown in equation 4.13.

$$\Delta = \begin{array}{cccc} S_1 & S_2 & S_3 & \\ 0 & 1 & -1 & S_1 \\ & 0 & -2 & S_2 \\ & & 0 & S_3 \end{array} \tag{4.13}$$

2. The A matrix is constructed as shown in equation 4.14.

$$\begin{bmatrix} -l_1 & l_2 & 0 \\ -l_1 & 0 & l_3 \\ 0 & -l_2 & l_3 \\ l_1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} = Al \tag{4.14}$$

3. L^* is then constructed form the time delay matrix (Δ). This is shown in equation 4.15.

$$L^* = \begin{bmatrix} \Delta_{12} \\ \Delta_{13} \\ \Delta_{23} \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ -2 \\ 0 \end{bmatrix} \tag{4.15}$$

4. Solving $l = A^{-1}W^{-1}WL^*$ yields l (refer to equation 4.16), assuming $W = I$ for this example. This shift is also shown in figure 4.2.

$$l = \hat{l} = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix} \tag{4.16}$$

   where $l = \hat{l}$ in this case.

The results of this algorithm – using the synthetic and Tennessee Eastman data sets – is discussed in chapter 6.

CHAPTER 5

Results: Naïve Clustering

The clustering algorithms defined in section 3.2 are applied to various data sets. We define benchmarks for perfectly separate clusters and totally random values. We then compare our synthetic data set and the Tennessee Eastman data set to these and quantify the performance of the clustering algorithms on a normal time series data set.

5.1 Synthetic data

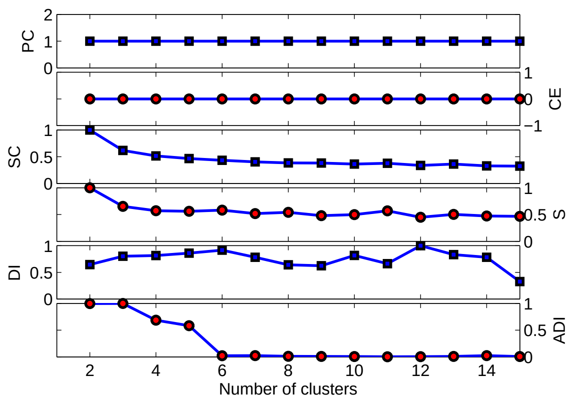
5.1.1 Optimal number of clusters

The number of clusters should be known before the partitioning. This is seldom the case as the underlying structure of the data is not known. To identify the optimal number of clusters, metrics for values of c between 2 to 15 were compared.

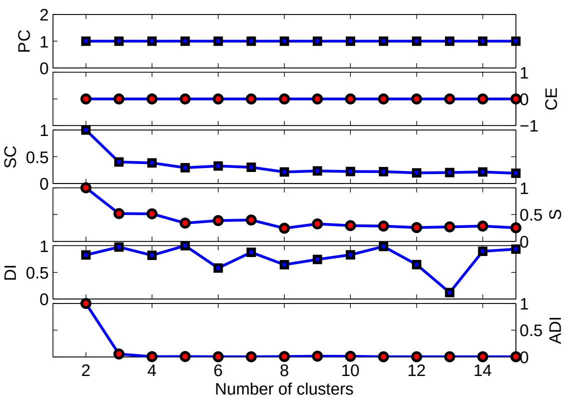
Because no validation index is reliable by itself, they should be compared with one other in order to find the optimal number of clusters. The actual values of the indices are not of importance at this point and therefore they were normalised with respect to the largest value of each. This gives a qualitative perspective of the optimal number of clusters.

Figures 5.1(a) and 5.1(b) show the validation indices when the K-means and K-medoid algorithms are applied to the synthetic data set.

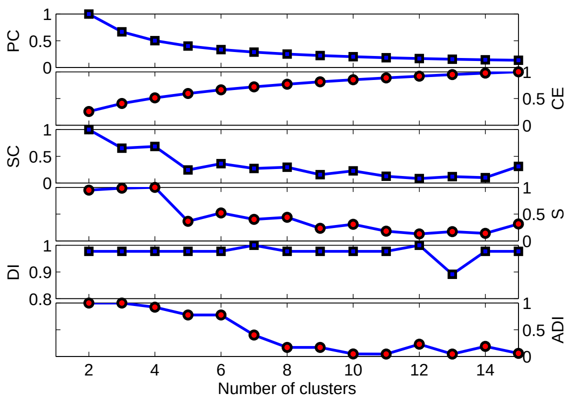
It is clear that no optimal number of clusters exist for the K-means and K-medoid algorithms. As mentioned in section 3.3, the PC and CE indices are 1 and 0 respectively when the partitioning is hard. This indicates a high probability that neither the K-means nor K-medoid algorithms can find substructures within the data. For this reason, no conclusion can be made regarding the quality of the clustering. The Partition Index (SC)



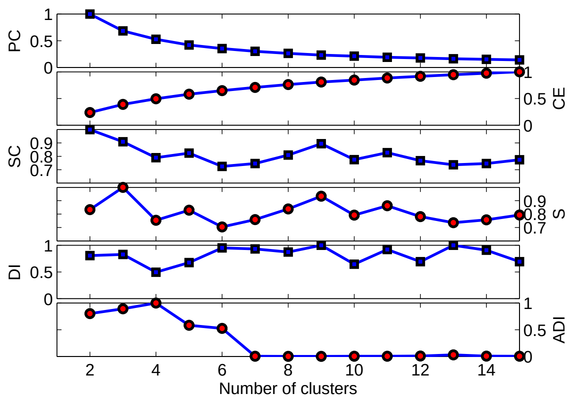
(a) K-means



(b) K-medoid



(c) Gustafson-Kessel



(d) Fuzzy C-means

Figure 5.1: Validation indices for the optimal number of clusters using the using the K-means, K-medoid, Gustafson-Kessel and Fuzzy C-means algorithms, applied to the Synthetic data set. The values are normalised with respect to the maximum value.

and Separation Index (S) decrease monotonically but S seems to flatten after $c = 4$, which indicates that there is little improvement in the ability of the clustering algorithm to identify well separated substructures within the data. Dunns Index (DI) fluctuates right through the cluster range for K-means. The alternative Dunn Index (ADI) shows a sharp drop when $c \geq 6$. This would suggest that the minimal distance between the clusters is now sufficiently small while the dispersion between data points is large, resulting in a near zero.

Figure 5.1(c) shows the result for the validation indices when the Gustafson-Kessel algorithm is used. Here the lack of direct connection to the data structure for both PC and CE is evident . They are monotonically increasing and decreasing respectively (Xie & Beni, 1991). Therefore, no information regarding a possible optimum number of clusters can be derived from them. The remaining indices all fluctuate through the cluster range. As with figures 5.1(a) and 5.1(b), we find SC and S decrease with an increase in c (there are slight upward tendencies at higher values, but the overall trend is decreasing). This also suggests that the clustering algorithm, Gustafson-Kessel in this case, struggles to identify unique substructures within the data. The total separation of the clusters increases, where the variation within the clusters decrease as c increases. This is more prominent in figure 5.4. Both the DI and ADI yield lower values at large numbers of cluster centres. These are not feasible solutions, as the model will become complex and will not result in anything meaningful.

As with the GK algorithm, the PC and CE indices show no clear optimum as they are monotonically decreasing and increasing respectively. As with all the previous clustering algorithms, the SC and S indices show a decreasing tendency with intermittent spikes as c increases. The values for all the other indices follow the same trend as shown in figure 5.1(c), with the ADI being the exception. It falls away completely at $c \geq 7$. This result aligns it with the K-medoid algorithm.

The GG algorithm is very unstable and difficult to implement. This is due to the small distances, $|\mathbf{x}_k - \mathbf{v}_i|$, used in equation 3.39 combined with the inversion of this equation. This algorithm could not be used on more than 2 cluster centers without running into numerical problems, so it will not be discussed further.

From these results it is clear to see that there is no conclusive optimal number of clusters for this data set. The internal structure of the data set is complex due to the time dependency of the data, so algorithms cannot separate different operating regions. As mentioned in section 3.3, Singhal & Seborg (2005) developed a method to determine the optimal number of clusters using of a pre-defined cost function. In their case they defined a cluster dissimilarity cost function, but for the purposes of this project, the cost functions would be the same those as discussed in sections 3.2.1 to 3.2.4.

Figure 5.2 shows the resultant “knee” plot obtained by evaluating the second derivative of the cost functions.

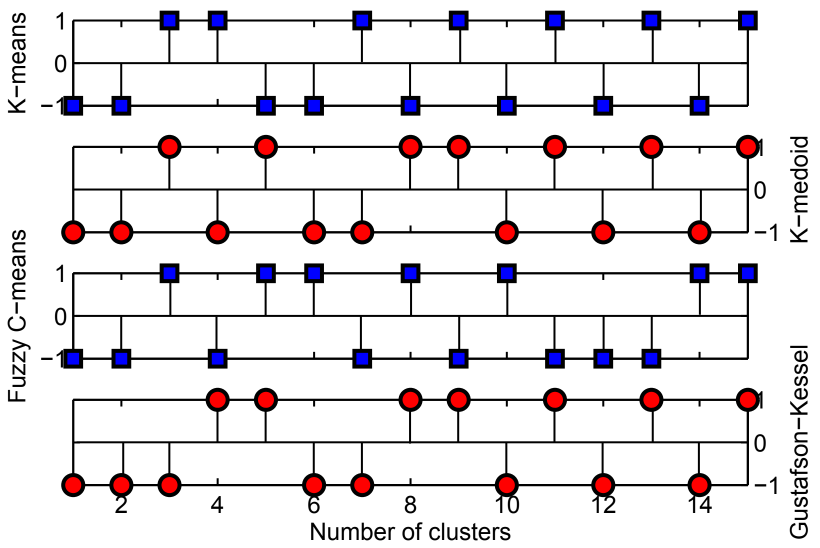


Figure 5.2: Second derivative 'Knee' plot from the cost functions of the respective clustering algorithms.

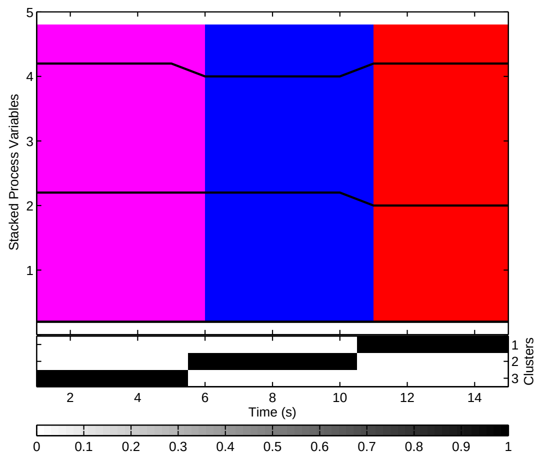
Singhal & Seborg (2005) states that the location of the first "knee" in this series (where the sign changes from negative to positive) is a good indication of the optimum number of clusters to be used for the particular clustering algorithm. The optimum for all the clustering algorithms except GK is at $c = 3$, where GK is at $c = 4$. The combination of the validity indices, from which no conclusive optimum could be found, with the method described by Singhal & Seborg (2005), yields tangible results and for the remainder of this section, the number of clusters to be used in each clustering algorithm would be same as mentioned above.

5.1.2 Results with a fixed number of clusters

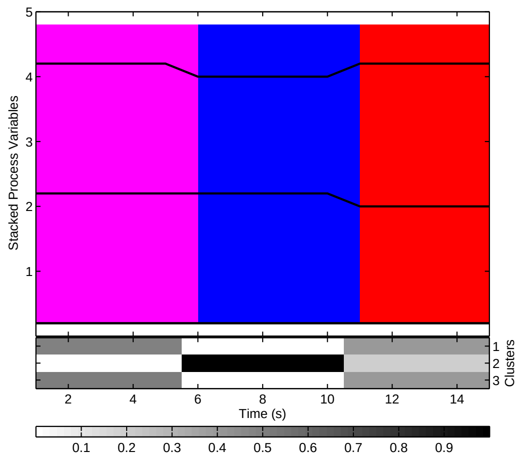
Validity Indices

To establish a base case to which our results can be related, we created two base case data sets.

1. A time series data set which contains 3 signals with no noise as well as 3 pre-defined clusters. This will be the best case scenario. Figure 5.3(a) shows the result of the K-means, K-medoid and Fuzzy C-means clustering algorithms. Figure 5.3(b) shows the result of the Gustafson-Kessel algorithm. It is clear that the algorithms find the three clusters. The Gustafson-Kessel algorithm however struggles to find 3 unique clusters. This result is unexpected as the Fuzzy C-means algorithm found the three clusters without any problems. The resulting validity indices are shown in table 5.1.



(a) K-means, K-medoid and Fuzzy C-means



(b) Gustafson-Kessel

Figure 5.3: Time series clusters for "best case" data set. The color bar at the bottom indicates the degree to which cluster 1 to 3 belongs to a certain time.

Table 5.1: Validity indices for various clustering algorithms using a random data set.

	K-means	K-medoid	Fuzzy C-means	Gustafson-Kessel
PE	1	1	0,62	1
CE	0	0	0,58	0
SC	0	0	0	0
S	0	0	0	0
DI	inf	inf	inf	inf
ADI	0	0	0	0

This result agrees with the expected result when the clusters are extremely well separated with no overlapping and no variance within the cluster itself.

2. A time series set of 9 random signals with the same mean and variances as the synthetic data defined in section 2.1. This will be the worst case scenario as all the signals are uncorrelated. Table 5.2 shows the result for the random data. We

Table 5.2: Validity indices for various clustering algorithms using a random data set.

	K-means	K-medoid	Fuzzy C-means	Gustafson-Kessel
PE	1	1	0,333 3	0,333 3
CE	0	0	1,098 6	1,098 6
SC	1,077 5	0,687 4	104 514 945	33 049 295
S	0,001 1	0,000 9	128 582	40 915
DI	0,110 2	0,086 9	0,09	0,093 6
ADI	0,006 1	0,008 3	0,011	0,013 5

see excessive values when comparing the SC and S indices to the other indices, for the fuzzy clustering techniques. The reason for these high values is due to the small separation of the clusters as well as the high variance within each cluster, as indicated by equations 3.48 and 3.49. The fuzzy algorithms cannot separate the data points into different clusters, and for this reason, the cluster centers are very close to each other. These values are substantially smaller for the hard partitioning algorithms when compared to the fuzzy results. The values for PC and CE give no indication of cluster goodness for the hard partitions, as they are at their maximum and minimum values respectively. This would suggest a lack of structure within the data. The fact that these values decrease and increase monotonically respectively, when using fuzzy clustering techniques, makes them of little use in the interpretation (Pal & Bezdek, 1995).

Table 5.3 compares the validity indices obtained by the different clustering algorithms when a noise-free synthetic time series data set is used.

When comparing the indices for the K-means (figure 5.1(a)) and K-medoid (figure 5.1(b)) algorithms, we see that the PC and CE indices are the same (1 and 0 respectively). This gives no indication which algorithm is superior in terms of identifying separate clusters. The SC, S and DI indices show that the K-medoid algorithm is superior to the K-means algorithm. The ADI shows that the K-means algorithm performs better than the K-medoid algorithm, but this result is ignored as the 3 indices mentioned earlier show otherwise. It is also worth mentioning that the PE and CE for the Gustafson-Kessel indices are the same for the synthetic data and the random data sets. The validity indices

Table 5.3: Validity indices for various clustering algorithms using noise free data.

	K-means	K-medoid	Fuzzy C-means	Gustafson-Kessel
PE	1	1	0,389 3	0,333 3
CE	0	0	1,013 5	1,098 6
SC	0,297 7	0,257 5	1,031 2	420 582
S	0,000 3	0,000 4	0,001 4	669,21
DI	0,096 6	0,069 2	0,082	0,067 2
ADI	0,018 9	0,002 0	0,032 9	0,047 4

for the Gustafson-Kessel algorithm (table 5.3) shows a marked spike in both the SC and S indices. This result will become apparent in figure 5.4, where the algorithm cannot find a suitable set of clusters for the data.

These values agree with those of the random data set. This suggests that the time series data is equivalent to random noise for the Gustafson-Kessel algorithm. The indices for the Fuzzy C-means algorithm are substantially better than that of the Gustafson-Kessel algorithm. This is the same when comparing it to the random data set.

We find that most of the clustering algorithms are at least able to find some structure in the synthetic time series plot, when compared to random time series values. Some of the validity indices are questionable. The ADI performs better in the case of the random data set, when the fuzzy algorithms are used and does not represent the truth as we know it should perform worse. The PE and CE indices also offer little value, as they are either 1 or 0, or decreasing and increasing monotonically. For these reasons, we are discarding these indices in the work to come.

Visual Clustering

The clustering algorithm, combined with a suitable dimensional reduction technique, should provide the user with a clear separation of the data clusters. The dimensional reduction performance values represented for each clustering algorithm and the respective dimensional reduction techniques are shown in table 5.4.

When $p = \|U - U^*\|$ is small, the dimensional reduction technique is considered to be good. This is because the difference between U (the original distances between the data points and the cluster centers) and U^* (the distances between the projected data and the projected cluster centers) is small.

The fuzzy Sammon mapping performed the worst except when using the hard partition clustering algorithms. It is evident from table 5.4 that PCA performed the best, while the Sammon mapping came second. This is also visible in figure 5.4, where a clear distinction between the different clusters is seen with minimal overlap, while using the K-means algorithm. The K-medoid algorithm does show some overlap and this is confirmed by the

Table 5.4: Performance indicators for various clustering and dimensional reduction algorithms.

Clustering	Dim. Red.	$p = \|U - U^*\|$	$\sum_{k=1}^{N} \overline{\mu_k^2}$	$\sum_{k=1}^{N} \overline{\mu_k^{2*}}$
K-means	PCA	0,151 7	1,000 0	0,663 9
	Sammon	0,170 9	1,000 0	0,630 5
	Fuzzy Sammon	0,238 4	1,000 0	0,591 5
K-medoid	PCA	0,226 5	1,000 0	0,595 3
	Sammon	0,212 3	1,000 0	0,608 4
	Fuzzy Sammon	0,285 0	1,000 0	0,506 3
Fuzzy C-means	PCA	0,098 1	0,389 3	0,515 5
	Sammon	0,087 2	0,389 3	0,497 3
	Fuzzy Sammon	0,040 6	0,389 3	0,410 8
Gustafson-Kessel	PCA	0,000 2	0,333 3	0,333 3
	Sammon	0,000 2	0,333 3	0,333 3
	Fuzzy Sammon	0,000 1	0,333 3	0,333 3

S index in table 5.3, where it performed slightly worse than the K-means algorithm.

The Fuzzy C-means algorithm produces three distinct cluster lobes. We see similar values for all three dimensional reduction techniques. This is also evident in figure 5.4, where the differences between different techniques are negligible.

The Gustafson-Kessel algorithm performs the worst of the clustering algorithms. This is evident from both figure 5.4 and table 5.3 where no meaningful clusters could be detected with large values in both the SC and S indices. This algorithm scores so well with respect to the dimensional reduction (table 5.4) because there is little difference between the distances in the higher and reduced dimensional spaces.

It is clear that the time-series data produces spherical clusters, which aids the performance of the hard partitioning cluster algorithms. This negates the possible advantages of the fuzzy algorithms where they could detect clusters of various shapes.

Figures 5.5(a) to 5.5(d) represent the same information as that in figure 5.4. These figures show the clusters on a time series plot. It is important to note that in the Fuzzy C-means and Gustafson-Kessel cases, the resulting partition is fuzzy. For this reason, a specific data point in time cannot be allocated to only one cluster, as it belongs to every cluster (with varying percentages). Therefore, the shaded areas only represent the cluster with which a specific data point is most associated.

From a practical point of view, it seems that the K-means algorithm results in the most consistent time series representation when the different shading is observed. The clusters seem to occur with minimal interruption from other clusters. This could be a contributing factor when we look at the K-means PCA combination in figure 5.4, where the different states of the signals are clearly clustered. We see that the active cluster plot

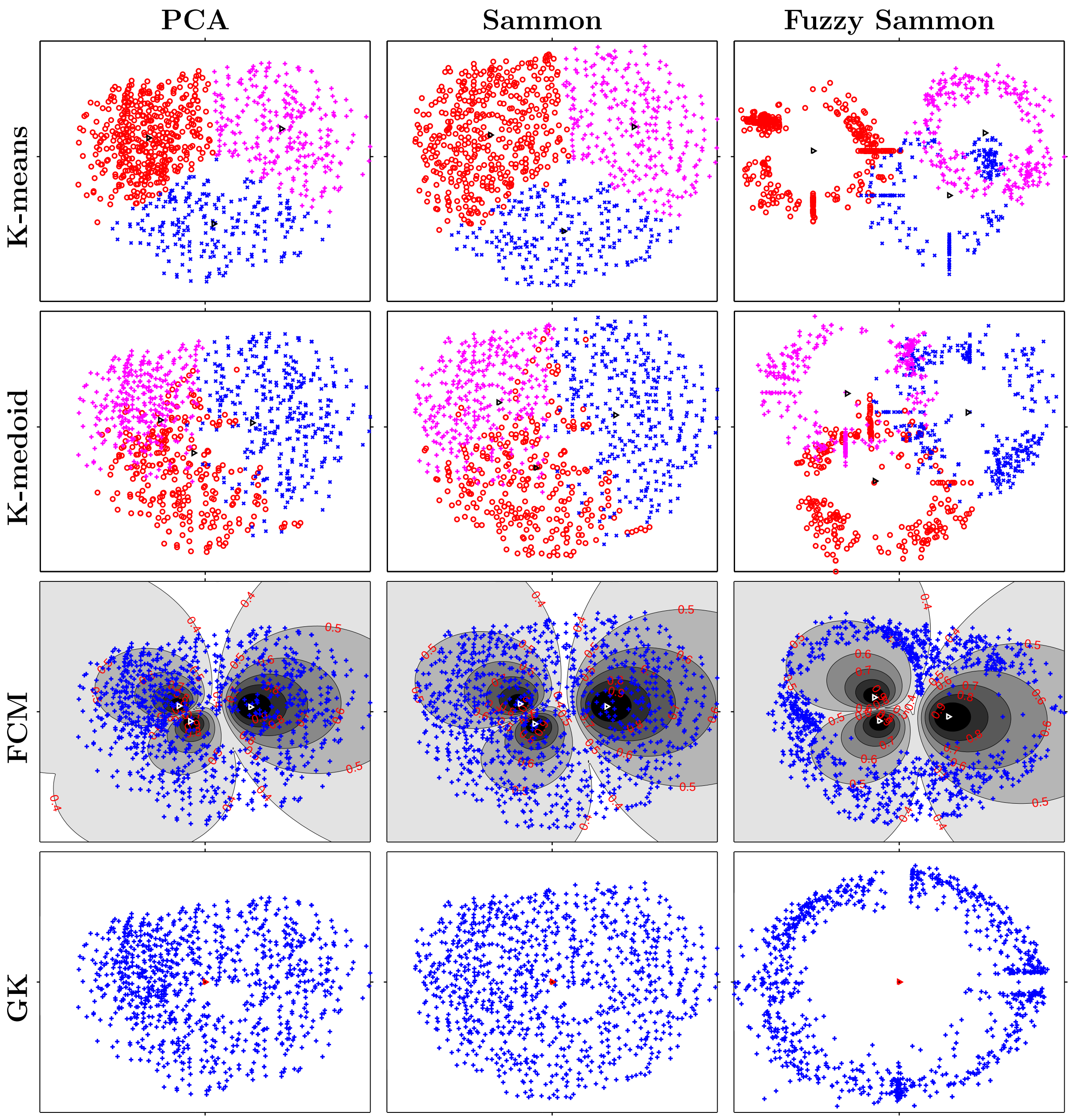
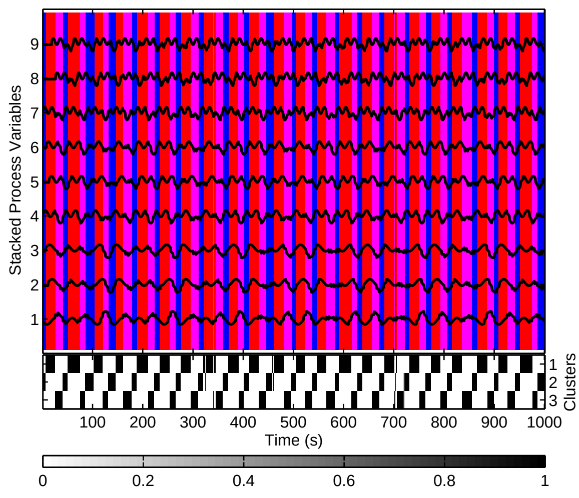
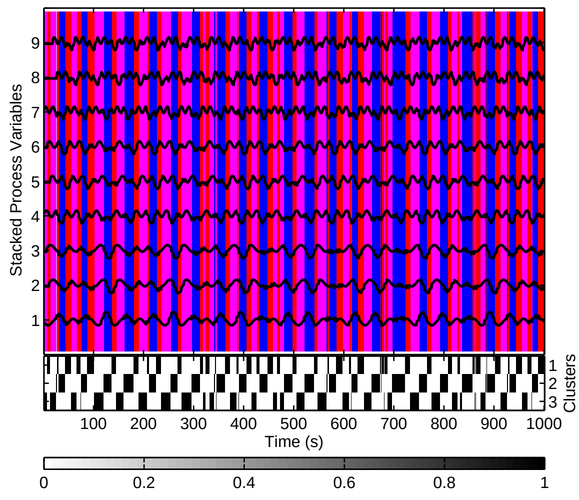


Figure 5.4: Visual representation of the K-means (top row), K-medoid (second from top), Fuzzy C-means (second from bottom) and Gustafson-Kessel (bottom row) algorithms. The columns represent the different dimensional reduction techniques, starting with PCA (first column), Sammon mappings (second column) and fuzzy Sammon mappings (third column). All the axes are normalised from -1 to 1, with the tick on each axis indicating zero. Four clusters centers were used in all but the Fuzzy C-means algorithm. These are indicated by the sideways triangles.

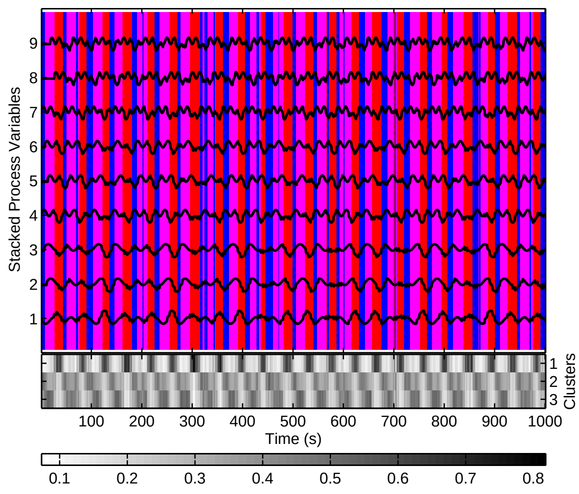
below the time series plot shows a definite set of stairs throughout the time range with respect to the active clusters. In other words, there is a certain number of continuity, or cyclical behaviour, with respect to the sequence in which the clusters are active. It starts at cluster no. 2, then works its way up to cluster no. 1, where after the sequence starts over again at 3. This is not a 100% consistent right through, but this would suggest that the clustering algorithm identifies the reoccurring time events.



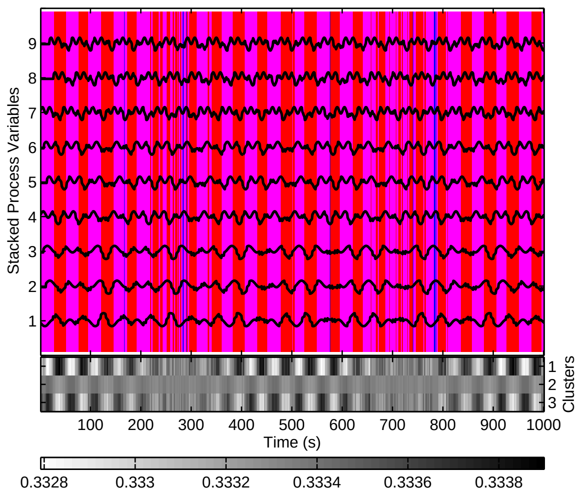
(a) K-means



(b) K-medoid



(c) Fuzzy C-means



(d) Gustafson-Kessel

Figure 5.5: Time representation of the different clustering algorithms using the Synthetic data set. The solid lines on the main plot represent the 9 process signals. They are shaded with different shades of gray to indicate to which cluster they belong at that point in time. The partial figure at the bottom indicates which cluster is "active" at that specific point in time.

The time series clusters of the K-medoid algorithm, as shown in figure 5.5(b), are more irregular than those seen in figure 5.5(a). The continuity of these clusters are not as good as in the previous case, resulting in a large number of small time increment clusters that belong to a given time. This translates to clusters that overlap as is evident in figure 5.4.

We find a similar pattern when looking at the time cluster plot of the Fuzzy C-Means algorithm, as shown in figure 5.5(c). The shaded areas only represent the cluster to which a data point belongs most to in with respect to time. The "active" cluster plot does however show the relative percentages to which a data point at a specific point in time belongs to a cluster. The fuzziness of the partition could be the cause of the lower scores in table 5.3 when compared to the K-medoid algorithm. The variance between the data in a cluster seems to be larger when these two are compared in figure 5.4, resulting in larger values for the SC and S indices. We also find that cluster 2 seems under-represented.

Figure 5.5(d) shows different clusters are active at different times in the figure. This should however be contrasted with the actual values of the "active" cluster plot, which does not vary significantly. This confirms the result obtained in figure 5.4, where no clear clusters formed. The reason for this behaviour is not clear. No meaningful information can be extracted form figure 5.5(d).

The K-means and K-medoid clustering algorithms provided the best results when clustering a synthetic time series data set, therefore, the Fuzzy C-means and Gustafson-Kessel algorithms are not investigated further. The PE, CE and ADI indices were found lacking (for various reasons), and are also discarded for the remainder of this project.

5.1.3 Influence of noise on clustering performance

In the previous section, the data were free of any noise. The clustering results of the random values data set showed a definite decrease in the performance of the clustering algorithms.

The noise power (height of the power spectrum density (PSD)) was increased from 0,001 to 10 in factors of 10. It is important to note that each signal had its own source of noise and these in turn had different "seed" values. Figure 5.6 show the results of the remaining validity indices. The K-means algorithm's validity indices show a steady increase up to a power of 10^{-1}, where after it reduces again. The effect of noise up to 10^{-1} decreases the ability of the clustering algorithm to obtain structure within the data. This is an intuitive result. However, it is counter intuitive to find the indices reduce in magnitude after a power of 10^{-1}. It stands to reason that high values for noise explicitly introduce structure within the data, but this cannot be confirmed.

The K-medoid clustering algorithm seems less sensitive to the effects of noise, as both SC and S reduce as the noise increases. The DI does however increase, but only marginally.

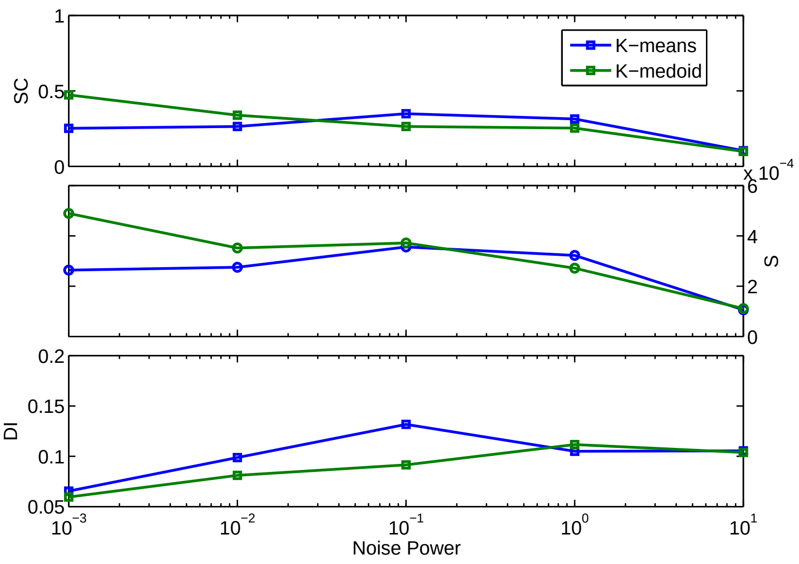


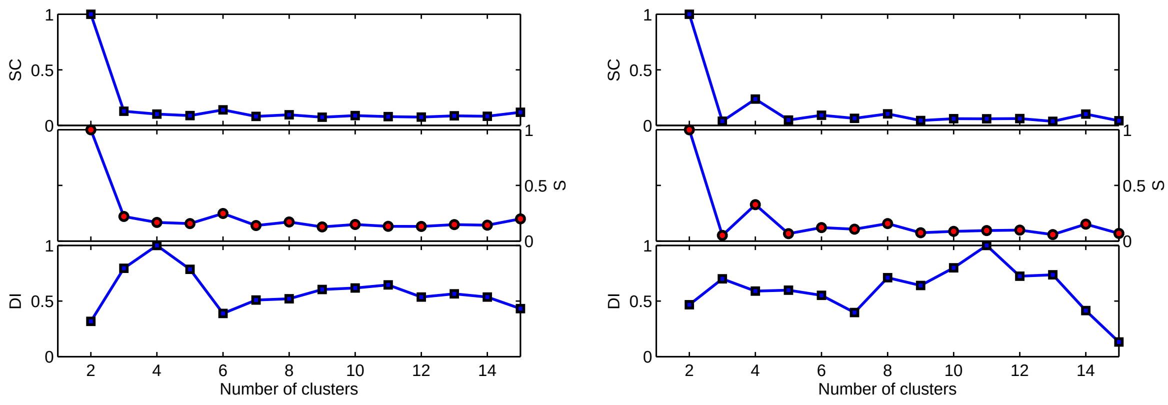
Figure 5.6: The influence of noise on clustering performance

5.2 Tennessee Eastman Data

5.2.1 Optimal number of clusters

As with the synthetic data set, the number of clusters were varied from 2 to 15 in a step wise fashion.

Figure 5.7 shows the results for the validation indices when the K-means and K-medoid algorithms are applied to the Tennessee Eastman data set.



(a) K-means (b) K-medoid

Figure 5.7: Validation indices for the optimal number of clusters using the K-means and K-medoid algorithms applied to the Tennessee Eastman data set. The values are normalised with respect to the maximum value.

As in section 5.1.1, the normalised values for the indices yield contradictory results with respect to the optimal number of clusters. In figure 5.7(a), the SC and S indices show a sharp decline at $c = 3$, then remain relatively constant. The DI shows a minimum at $c = 2$, indicating a possible optimum. A similar situation occurs when the K-medoid algorithm is used. The SC and S indices obtain minima at $c = 3$ while DIs minimum occurs at $c = 15$, but this is not conclusive result, as the number of clusters is limited to 15. The internal data structure is much more complex than that of the synthetic data set. There are many more transient states, which blur the distinction between different operating regions. The knee plot, shown in figure 5.8, suggests that the optimum occurs at $c = 4$. This is comparable with the optimal number of clusters obtained with the validation indices.

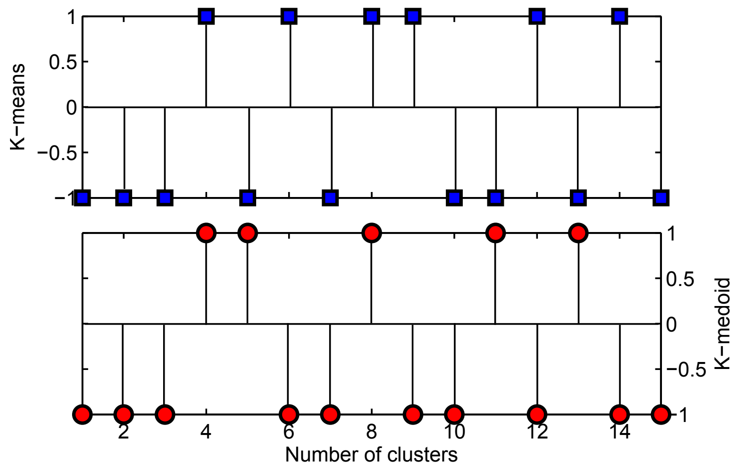


Figure 5.8: Second derivative "Knee" plot from the cost functions of the respective clustering algorithms using the Tennessee Eastman data.

5.2.2 Results with a fixed number of clusters

In the previous section, the optimal number of cluster centers were found to be $c = 4$. These results are implemented in this section.

Validity indices

Table 5.5 shows the reduced number of validity indices for both the K-means and K-medoid clustering algorithms.

All three the indices show similar values for both the clustering algorithms. The SC and S indices are substantially smaller than those obtained from the synthetic data set (refer to table 5.3), While the DI is marginally higher. In this case, the cluster separation is larger than in the case of the synthetic data set (in the higher dimension). Although

Table 5.5: Validity indices for various clustering algorithms while using the Tennessee Eastman data.

	K-means	K-medoid
SC	0,074 6	0,084 4
S	0,000 1	0,000 1
DI	0,139 8	0,083 5

the variance could be higher, the actual cluster distances offset this and results in smaller values for both the SC and S indices.

Visual Clustering

Figure 5.9, shows the clustering results for the Tennessee Eastman data set. It is immediately evident that the Sammon and Fuzzy Sammon mappings perform the worst.

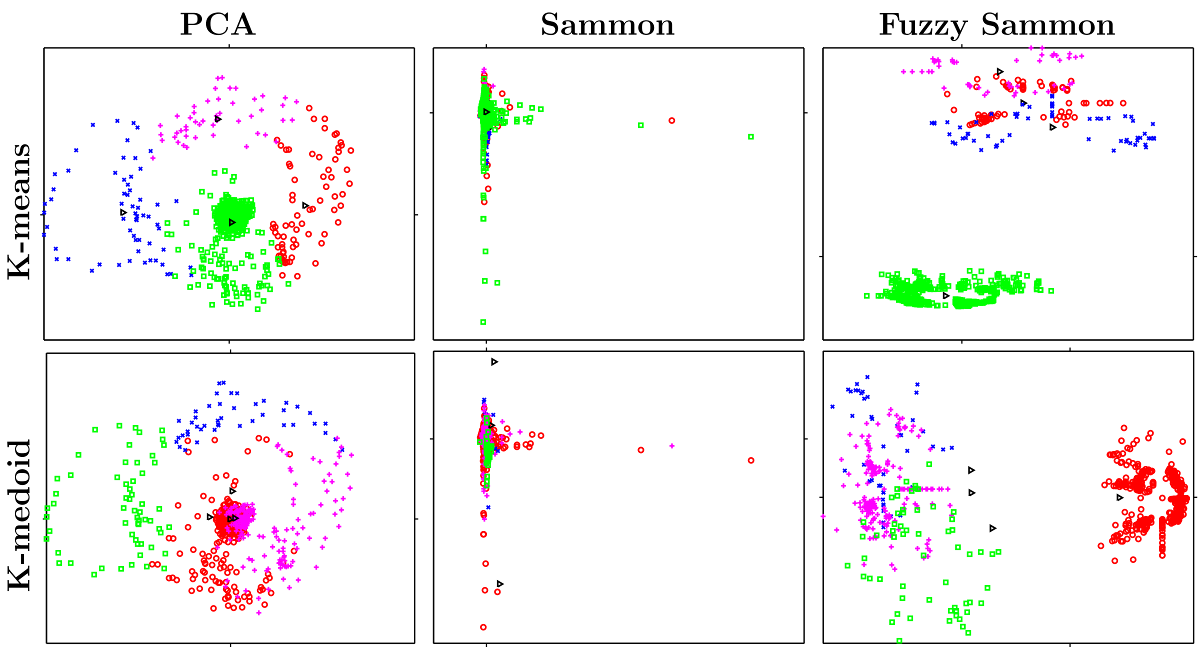


Figure 5.9: Visual representation of the K-means (top row) and K-medoid (second from top) algorithms. The columns represent the different dimensional reduction techniques, starting with PCA (first column), Sammon mappings (second column) and fuzzy Sammon mappings (third column).All the axes are normalised from -1 to 1, with the tick on each axis indicating zero. Four clusters centers were used. These are indicated by the sideways triangles.

The Fuzzy Sammon mapping succeeds in separating one cluster from the rest, for both the K-means and the K-medoid clustering algorithms. From figures 5.10(a) and 5.10(b), it is evident that these are the largest clusters in the time series. However, they arent able to distinguish between the smaller clusters, making this technique redundant. Table 5.6

does suggest that the Fuzzy Sammon mapping yields a better result in terms of preserving the original projection with respect to the other dimensional reduction techniques, as the difference between the original and projected distances are very small (as it forces the lower dimensional inter point distances to be the same as those in the higher dimension). When these results are compared to those in table 5.4, we find that the Fuzzy Sammon mapping performs better for the more complex Tennessee Eastman data set.

The Sammon mapping does not succeed in separating the data. It yields to a single group of data points with no way of distinguishing between the different clusters. This is also reflected in the relatively high values of P. The Sammon mapping performs worse when compared to the synthetic data set.

The PCA results in the best visual representation of the data in the lower dimensional space. The clusters are separated with minimal overlapping in this space. The results for the K-means algorithm are comparable with the more computationally expensive Fuzzy Sammon mapping. However, the K-medoid algorithm's PCA projection is not very good. It performs worse than in the case of the Synthetic data set (refer to table 5.4).

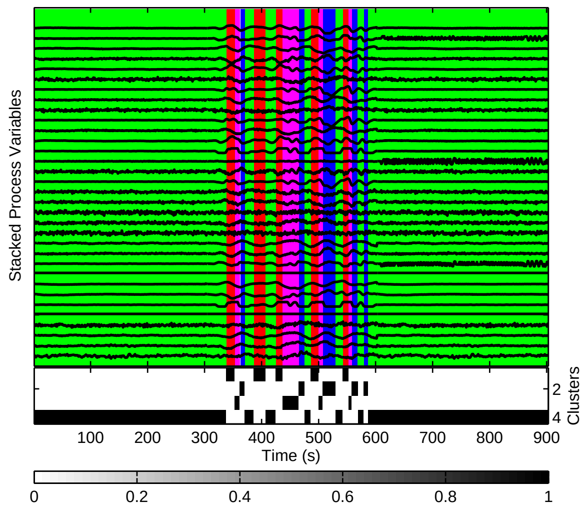
Table 5.6: Performance indicators for various clustering and dimensional reduction algorithms.

Clustering	Dim. Red.	$p = \|U - U^*\|$	$\sum_{k=1}^{N} \overline{\mu_k^2}$	$\sum_{k=1}^{N} \overline{\mu_k^{2*}}$
K-means	PCA	0,067 5	1	0,797 8
	Sammon	0,259 4	1	0,464 4
	Fuzzy Sammon	0,051 5	1	0,890 5
K-medoid	PCA	0,285 7	1	0,428 6
	Sammon	0,203 9	1	0,729 8
	Fuzzy Sammon	0,163 1	1	0,577 3

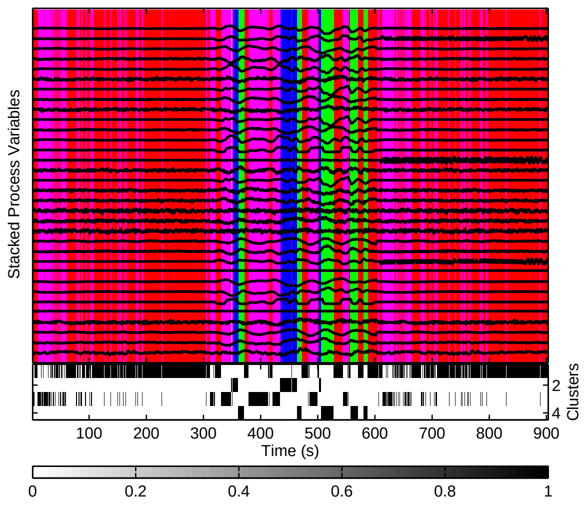
Figures 5.10(a) and 5.10(b) show the time series interpretation of figure 5.9.

The K-means algorithm identifies 2 operating regions. The first, and largest is represented by the green sections (or cluster). It is evident that these sections differ substantially from the middle section with respect to the shape of the signals. The three real regions in the data are not identified, but an operator would be aware of the different periods of operation.

The K-medoid algorithm yields a similar result than that of the K-means algorithm. The main difference lies in the first and last sections (the two green sections in the K-means result). Where the K-means algorithm was only able to distinguish these sections from the middle section, we see that the K-medoid algorithm imparts 2 clusters for these sections (red and magenta). From an assistive technology point of view, this would not help the operator to distinguish between different operating regions.



(a) K-means



(b) K-medoid

Figure 5.10: Time representation of the different clustering algorithms using the Tennessee Eastman data set. The solid lines on the main plot represent the process signals. They are shaded with different shades of gray to indicate to which cluster they belong at that point in time. The partial figure at the bottom indicates which cluster is “active” at that specific point in time.

CHAPTER 6

Results: Time Delay Estimation

We show that a simple weighting function can increase the performance of the algorithm when complex systems are used. The goal of this section is to calculate a single set of optimum shifts that would increase the performance of the clustering algorithm.

6.1 Synthetic Data

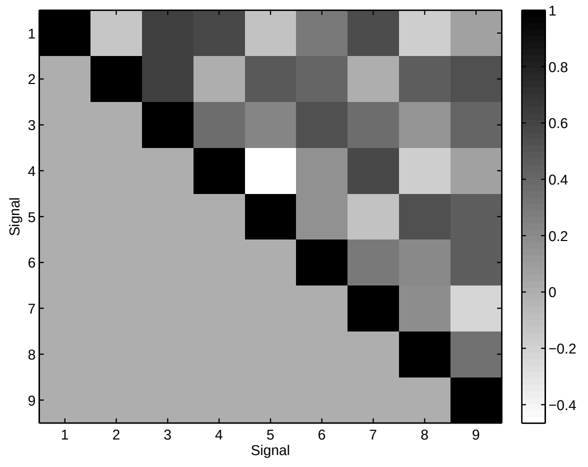
The synthetic data set consists of 3 groups of 3 signals. The time delays of these signals were selected in an arbitrary fashion. As discussed in section 4.1, the multiple signal time delay estimation (MSTDE) algorithm relies on both the correlation coefficients of the signals as well as the cross correlation time delays to estimate the true time delay of the system.

Figure 6.1(a) shows the correlation coefficients for the 9 signals.

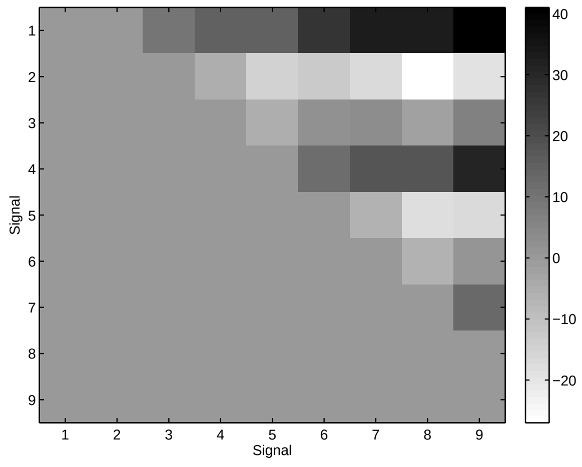
The diagonal elements are equal to 1, as the signal correlates 100% with itself over time. Because signals 1-3, 4-6 and 7-9 form groups in this set, we would expect some block diagonal structure to form off the diagonal, but this is not the case. Although the signals' frequencies are co-prime, there is still some correlation between them.

The results of the cross correlation lags are shown in figure 6.1(b). It is clear that there is a large variety of lags present in the system. These lags are related to one another by some form of causal structure, but we need to reduce it further to have any practical application. The results of the MSTDE algorithm is shown in table 6.1.

This results compare 100% with the actual pre-defined time delays. The algorithm identifies subgroups within the data. This is indicated by the 3 zero values in table 6.1. The norm of the residual is very small , $|Al - L^*| = 1{,}4 \times 10^{-14}$, which indicates that the result of the optimisation is consistent.



(a) Correlation Coefficients



(b) Cross Correlation Lags

Figure 6.1: Correlation coefficients and Cross-Correlation Lags for the synthetic data set.

Table 6.1: Results of the MSTDE algorithm applied to the synthetic data set.

Signal	1	2	3	4	5	6	7	8	9
Shift	0	10	5	0	12	6	0	13	9

When the correlation coefficients are used as weights in the optimisation algorithm, the results are similar. The optimal time shifts are the same as in table 6.1, but the norm of the residual is smaller, $|WAl - WL^*| = 4{,}9 \times 10^{-15}$.

The threshold weighting function (refer to section 4.1) also yields the same results as shown in table 6.1 with $|WAl - WL^*| = 1{,}05 \times 10^{-14}$.

Two different interpretations can result form table 6.1:

1. The algorithm suggests that signals 1, 4 and 7 are the base signals and the remainder are shifted. In other words, signals 2 and 3 will be shifted in relation with signal 1 and so on.

2. The algorithm does not find any significant correlation between signals 1, 4, and 7 to the rest of the data. Therefore, while the other signals are shifted to align themselves better with each other, signals 1, 4, and 7 are not shifted as they are unrelated.

This issue can be addressed by taking the correlation matrix into account to confirm the inter-signal uncorrelatedness.

Figure 6.2 shows the effect before and after the shift of the signals. The shifted signals are truncated after shifting to retain consistent start and end times for the data set.

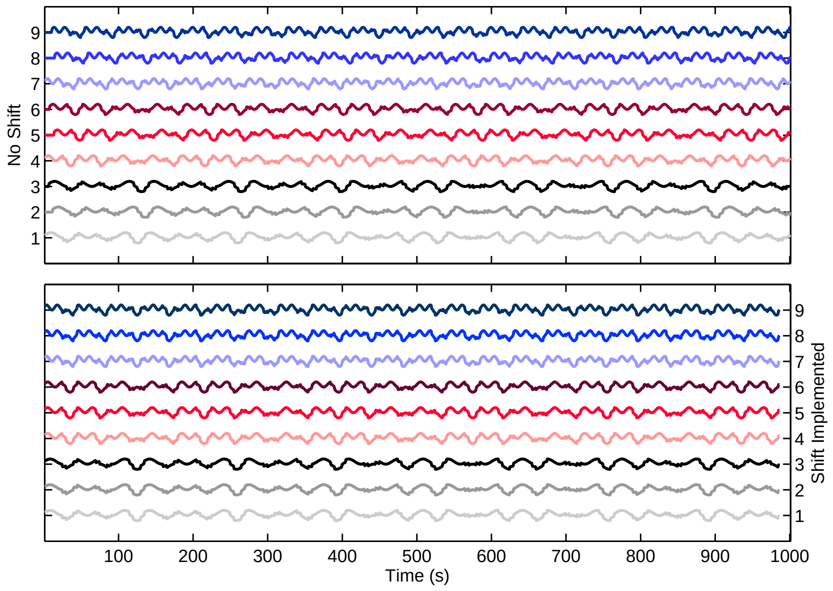


Figure 6.2: Results of time shifting the synthetic data set. The shifted signals align perfectly.

6.1.1 Effect of Noise in the MSTDE algorithm

Figure 6.3 shows the resiliency of the MSTDE algorithm to noise with the various weighting functions. It is clear that the "No weighting" solution has the largest residual norm. The norms for both the correlation coefficient weighting ("Corr Coef") and the Threshold weighting solutions are relatively small right through the range of noise[1].

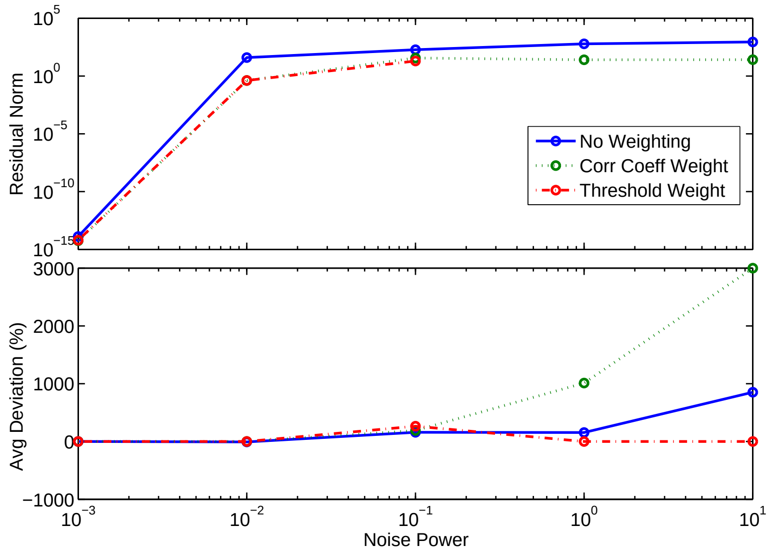


Figure 6.3: The norm of the residual as well as the average percentage fault per signal time delay.

[1]The residual norm cannot be compared between the different weighting functions, as it is application specific. However, it does give insight into the change of each weighting function norm over a noise range.

The average deviation was calculated by calculating the deviation from the known delay for all 9 signals. The average deviation is then reported at every noise power. It is counter-intuitive to find that the correlation weighting function made the largest gross errors in calculating the time delays when compared to no weighting at all. The threshold weighting function did not yield results above noise levels of 10^{-1}. The reason for this lies in the fact that the maximum correlation and correlation directionality values for the signals are all smaller than the threshold values. This essentially results in a weighting function of zero. This is a preferred result, as we do not want the algorithm to shift the signals if they are not statistically correlated.

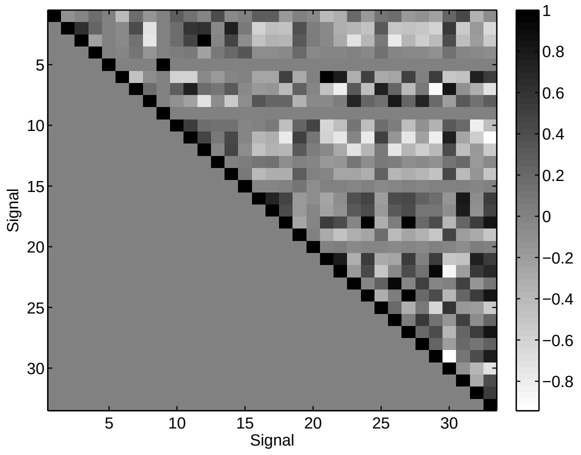
6.2 Tennessee Eastman Data

Because the Tennessee Eastman process is very complex, the threshold weighting method is used in the MSTDE algorithm. Figure 6.4(a) shows the correlation coefficients for the 33 signals described in table 2.3. It is evident that there are some highly correlated signals. We also see some blocks forming. These are most prominent at signal 5 (compressor recycle valve), signal 9 (stripper steam valve) and signal 20 (reactor temperature). This result is also visible in figure 6.4(b). The range of the calculated time delays is very large, and we see that the afore-mentioned signals contain the highest time delays. Bauer & Thornhill (2008) states that when using the cross-correlation function to determine the process lags, the user cannot expect useful lags larger than $\frac{N}{4}$, and this is indeed what has happened. Combined with the thresholding methods described in section 4.1, we find that these signals (and others) are neglected in the optimisation process by setting the associated weight equal to zero. Figure 6.4(c) shows the signals that are statistically valid (according to the threshold values).

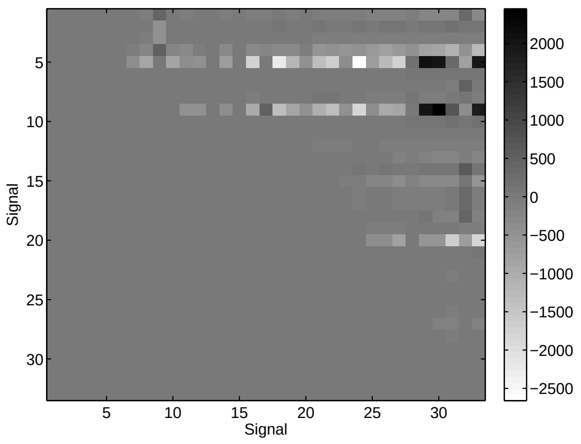
Table 6.2 shows the result of the MSTDE algorithm. We find relatively small time shifts compared to the magnitudes seen in figure 6.4(b). Because signal 9 (stripper steam valve) and signal 5 (compressor recycle valve) was not correlated with any other signals other than themselves in the correlation matrix (figure 6.4(a)), we do not expect a shift in any way. This is indeed the case and is shown in table 6.2.

Due to our prior knowledge of the construction of the data set, and the fact that it is a combination of three individual data sets, we would like to investigate them separately. The cross-correlation function will calculate a better time delay matrix for each subset of the large set due to the fact that there are different disturbances in the large data set. Each subset contains only one disturbance, and this would have beneficial effects on the calculated time shifts.

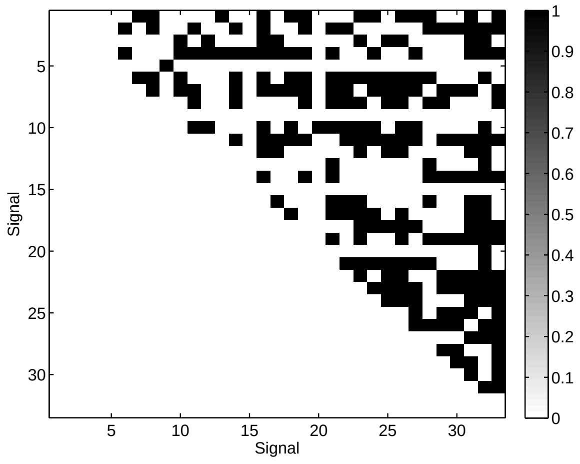
The first subset contains the IDV(3) disturbance (Step in D feed temperature). Figures 6.5(a) and 6.5(b) show the time delay matrix and the statistically valid signals respectively. The statistically valid signal matrix is much more sparse than in the previous



(a) Correlation Coefficients



(b) Cross Correlation Lags



(c) Statistically valid signals

Figure 6.4: Correlation coefficients and Cross-Correlation Lags for the synthetic data set.

Table 6.2: Results of the MSTDE algorithm applied to the Tennessee Eastman data set.

Variable Name	Variable number	Shift	Normalised Shift
Product separator level	XMEAS (12)	−9,3	0,0
Reactor cooling water outlet temperature	XMEAS (21)	−6,2	3,1
Stripper level	XMEAS (15)	−6,1	3,2
Reactor temperature	XMEAS (9)	−5,8	3,5
Reactor cooling water flow	XMV (10)	−5,8	3,5
Stripper pressure	XMEAS (16)	−5,4	3,9
Separator cooling water outlet temperature	XMEAS (22)	−5,4	3,9
Product separator pressure	XMEAS (13)	−5,3	4,0
Purge rate (stream 9)	XMEAS (10)	−5,2	4,1
Compressor work	XMEAS (20)	−5,1	4,2
Reactor pressure	XMEAS (7)	−5,1	4,2
Purge valve (stream 9)	XMV (6)	−4,9	4,4
E feed flow (stream 3)	XMV (2)	−4,8	4,5
E feed (stream 3)	XMEAS (3)	−4,7	4,6
Condenser cooling water flow	XMV (11)	−4,4	4,9
Reactor level	XMEAS (8)	−4,3	5,0
Stripper liquid product flow (stream 11)	XMV (8)	−3,6	5,7
Stripper underflow (stream 11)	XMEAS (17)	−3,6	5,7
Stripper temperature	XMEAS (18)	−3,4	5,8
Stripper steam flow	XMEAS (19)	−3,1	6,2
Product separator temperature	XMEAS (11)	−3,0	6,3
Reactor feed rate (stream 6)	XMEAS (6)	−2,8	6,5
Recycle flow (Stream 8)	XMEAS (5)	−2,4	6,9
Product separator underflow (stream 10)	XMEAS (14)	−1,8	7,5
Separator pot liquid flow (stream 10)	XMV (7)	−1,7	7,6
D feed (stream 2)	XMEAS (2)	−0,4	8,8
A feed (stream 1)	XMEAS (1)	−0,2	9,0
A feed flow (stream 1)	XMV (3)	−0,2	9,1
Compressor recycle valve	XMV (5)	0,0	9,3
Stripper steam valve	XMV (9)	0,0	9,3
D feed flow (Stream 2)	XMV (1)	0,0	9,3
A and C feed flow (stream 4)	XMV (4)	41,1	50,3
A and C feed (stream 4)	XMEAS (4)	41,1	50,3

case with blocks forming in the matrix. This will result in a larger residual norm, and this is indeed the case, with $|Al - L^*| = 44{,}65$. The time shift results of the MSTDE algorithm (with threshold weighting) is shown in table 6.3.

There are more unsifted signals (as indicated by the zero shift) than in the original data set. This is due to the sparse weighting function shown in figure 6.5(b), which indicates that the signals are not correlated to each other.

The second subset contains the IDV(13) disturbance (slow drift of the reaction kinetics). Figures 6.6(a) and 6.6(b) show the time delay matrix and the statistically valid signals respectively. As with the original data set, we find very large delays in the time delay matrix. They are associated with signals 5 (compressor recycle valve), 9 (stripper steam valve) and 15 (A and C feed [stream 4]). These signals are removed from the optimisation algorithm by the threshold weighting function. The time shift results of the MSTDE algorithm (with threshold weighting) is shown in table 6.4. The norm of the residual is smaller than in the IDV(3) case, being $|Al - L^*| = 31{,}65$. This is due to a less sparse weighting matrix.

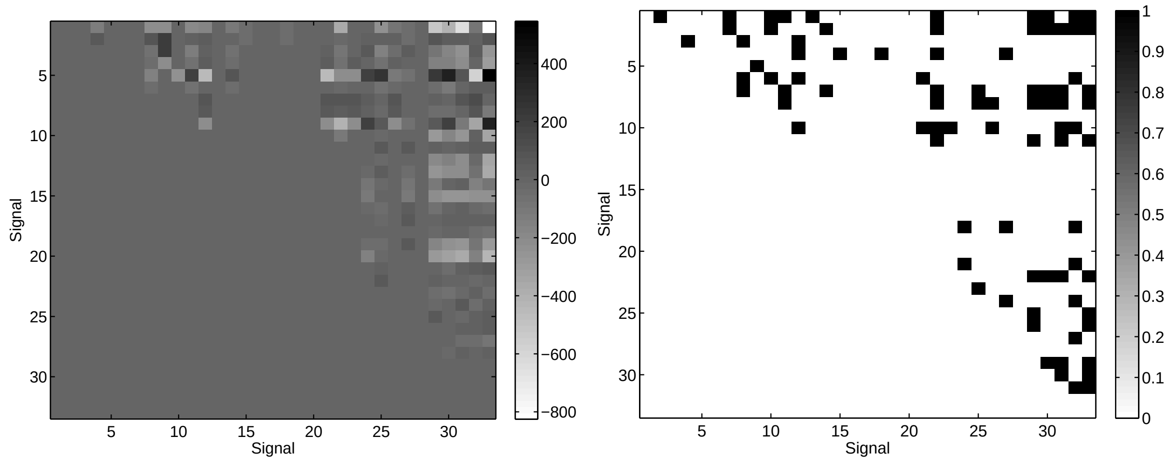
The third subset contains the IDV(14) disturbance (sticking reactor cooling water valve). Figures 6.7(a) and 6.7(b) show the time delay matrix and the statistically valid signals respectively. Although the statistically valid signal matrix is sparse (as in the IDV(3)) case, the norm of the residual is smaller, $|Al - L^*| = 35{,}07$. When these two matrices are compared, we find that the rank of the IDV(14) subset is 28 while that of the IDV(3) is 27. This could explain the difference in the resulting norm. The time shift results of the MSTDE algorithm (with threshold weighting) is shown in table 6.5.

6.3 Causality from time shifts?

Bauer & Thornhill (2008) developed the methods used in the previous sections (as described in section 4.1 as part of a causality inference system. It should be noted that this does not form part of the core of the project, but the unsubstantiated results will be highlighted in this section. For the purposes of this section, only the IDV(14) subset – sticky reactor cooling valve – will be considered.

Figure 6.8 shows the Tennessee Eastman process and can be read together with table 6.5 with regards to the propagation of a fault. We find that the reactor cooling water valve is at the top of the shift list, indicating that it may be the origin of the fault. This is followed by the general reactor region.

There are however some strange signals within this reactor region which seem out of place. They include the “Product separator pressure”, “Stripper pressure” and the ”Condenser cooling water flow”. Upon closer inspection (using figures 6.7(a) and 6.7(b)), the following becomes apparent:

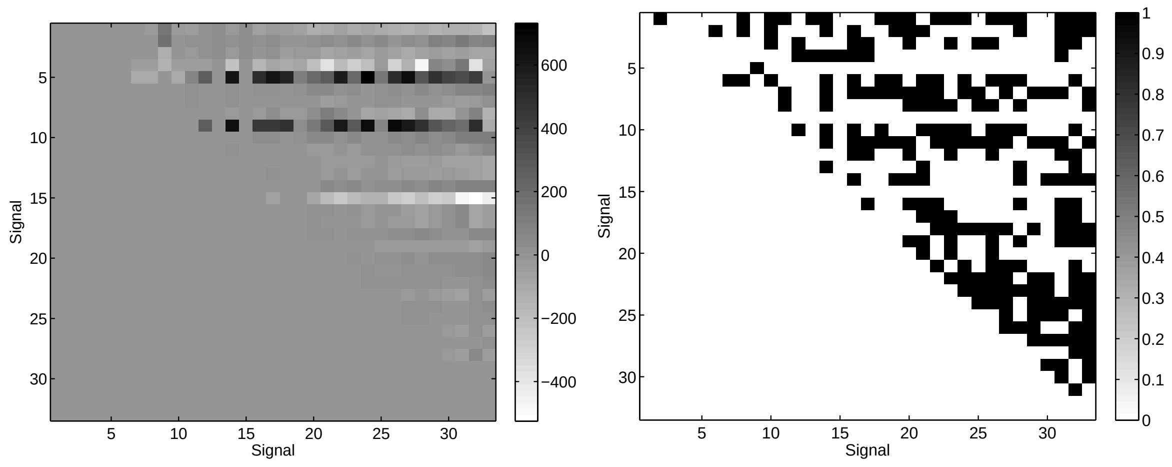


(a) Cross Correlation Lags (b) Statistically valid signals

Figure 6.5: Time delay matrix and statistically valid signals for the IDV(3) subset.

Table 6.3: Results of the MSTDE algorithm applied to the Tennessee Eastman data set with a step in the D feed temperature.

Variable Name	Variable number	Shift	Normalised Shift
Condenser cooling water flow	XMV (11)	−18,9	0,0
Compressor work	XMEAS (20)	−18,7	0,2
Separator cooling water outlet temperature	XMEAS (22)	−17,5	1,4
Product separator temperature	XMEAS (11)	−17,1	1,9
E feed (stream 3)	XMEAS (3)	−16,7	2,2
Stripper steam flow	XMEAS (19)	−16,2	2,7
Stripper temperature	XMEAS (18)	−16,0	2,9
Separator pot liquid flow (stream 10)	XMV (7)	−14,6	4,3
E feed flow (stream 3)	XMV (2)	−12,5	6,5
Product separator underflow (stream 10)	XMEAS (14)	−11,2	7,8
Stripper level	XMEAS (15)	−6,8	12,1
Reactor cooling water outlet temperature	XMEAS (21)	−5,6	13,3
A feed (stream 1)	XMEAS (1)	−5,6	13,4
A feed flow (stream 1)	XMV (3)	−5,4	13,6
A and C feed flow (stream 4)	XMV (4)	−5,3	13,6
A and C feed (stream 4)	XMEAS (4)	−5,3	13,6
Reactor pressure	XMEAS (7)	−5,1	13,8
Stripper pressure	XMEAS (16)	−5,1	13,8
Product separator pressure	XMEAS (13)	−5,1	13,8
Purge rate (stream 9)	XMEAS (10)	−3,7	15,3
Purge valve (stream 9)	XMV (6)	−3,7	15,3
Product separator level	XMEAS (12)	−3,2	15,7
Reactor cooling water flow	XMV (10)	−3,0	15,9
Stripper liquid product flow (stream 11)	XMV (8)	−0,9	18,0
Compressor recycle valve	XMV (5)	0,0	18,9
Stripper steam valve	XMV (9)	0,0	18,9
Recycle flow (Stream 8)	XMEAS (5)	0,0	18,9
Reactor feed rate (stream 6)	XMEAS (6)	0,0	18,9
Reactor level	XMEAS (8)	0,0	18,9
Reactor temperature	XMEAS (9)	0,0	18,9
Stripper underflow (stream 11)	XMEAS (17)	0,0	18,9
D feed flow (Stream 2)	XMV (1)	0,0	18,9
D feed (stream 2)	XMEAS (2)	0,0	18,9



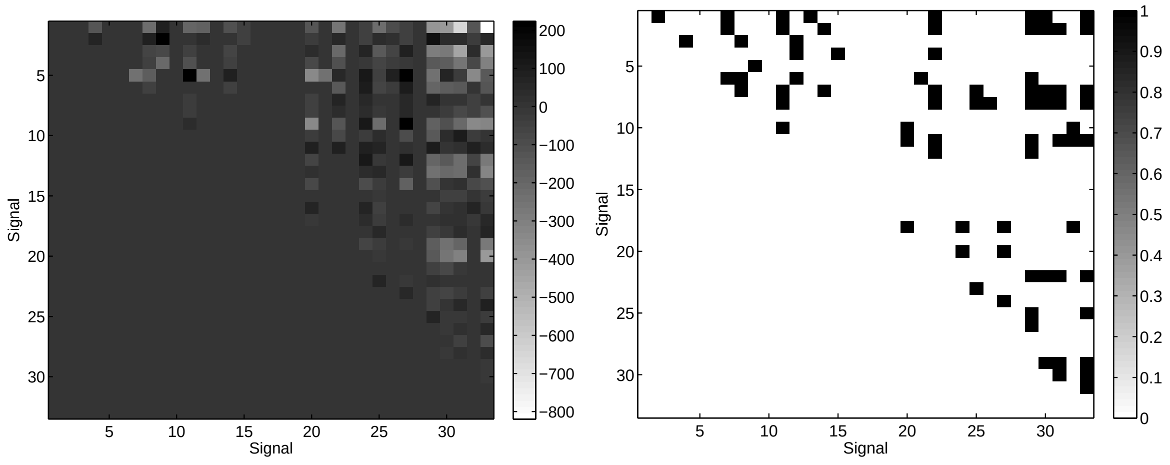
(a) Cross Correlation Lags

(b) Statistically valid signals

Figure 6.6: Time delay matrix and statistically valid signals for the IDV(13) subset.

Table 6.4: Results of the MSTDE algorithm applied to the Tennessee Eastman data set with slow drifting in the reaction kinetics.

Variable Name	Variable number	Shift	Normalised Shift
Reactor cooling water outlet temperature	XMEAS (21)	−6,0	0,0
Reactor cooling water flow	XMV (10)	−5,9	0,0
Product separator level	XMEAS (12)	−5,5	0,5
Reactor pressure	XMEAS (7)	−5,3	0,6
Product separator pressure	XMEAS (13)	−5,3	0,7
Stripper pressure	XMEAS (16)	−5,3	0,7
Separator cooling water outlet temperature	XMEAS (22)	−5,2	0,8
Compressor work	XMEAS (20)	−4,6	1,4
Purge rate (stream 9)	XMEAS (10)	−4,6	1,4
Reactor temperature	XMEAS (9)	−4,6	1,4
Purge valve (stream 9)	XMV (6)	−4,2	1,8
Condenser cooling water flow	XMV (11)	−4,1	1,9
Stripper temperature	XMEAS (18)	−3,9	2,1
Stripper level	XMEAS (15)	−3,5	2,5
Stripper steam flow	XMEAS (19)	−3,3	2,6
E feed (stream 3)	XMEAS (3)	−3,0	3,0
Product separator temperature	XMEAS (11)	−2,9	3,1
Stripper underflow (stream 11)	XMEAS (17)	−2,7	3,3
Recycle flow (Stream 8)	XMEAS (5)	−2,1	3,9
Reactor feed rate (stream 6)	XMEAS (6)	−2,1	3,9
Stripper liquid product flow (stream 11)	XMV (8)	−1,9	4,1
E feed flow (stream 3)	XMV (2)	−1,6	4,4
Product separator underflow (stream 10)	XMEAS (14)	−1,2	4,8
Reactor level	XMEAS (8)	−1,1	4,9
Separator pot liquid flow (stream 10)	XMV (7)	−0,9	5,1
D feed (stream 2)	XMEAS (2)	−0,9	5,1
Compressor recycle valve	XMV (5)	0,0	6,0
Stripper steam valve	XMV (9)	0,0	6,0
D feed flow (Stream 2)	XMV (1)	0,0	6,0
A feed (stream 1)	XMEAS (1)	1,6	7,6
A feed flow (stream 1)	XMV (3)	1,7	7,7
A and C feed (stream 4)	XMEAS (4)	30,0	35,9
A and C feed flow (stream 4)	XMV (4)	30,0	35,9



(a) Cross Correlation Lags (b) Statistically valid signals

Figure 6.7: Time delay matrix and statistically valid signals for the IDV(14) subset.

Table 6.5: Results of the MSTDE algorithm applied to the Tennessee Eastman data set with a sticky reactor cooling water valve.

Variable Name	Variable number	Shift	Normalised Shift
Reactor cooling water flow	XMV (10)	−25,4	0,0
Reactor temperature	XMEAS (9)	−25,4	0,0
Product separator pressure	XMEAS (13)	−25,4	0,0
Reactor pressure	XMEAS (7)	−25,4	0,0
Reactor cooling water outlet temperature	XMEAS (21)	−25,4	0,0
Condenser cooling water flow	XMV (11)	−25,4	0,0
Stripper pressure	XMEAS (16)	−25,4	0,0
Compressor work	XMEAS (20)	−25,0	0,3
Separator cooling water outlet temperature	XMEAS (22)	−22,3	3,1
E feed (stream 3)	XMEAS (3)	−21,1	4,3
Stripper steam flow	XMEAS (19)	−20,2	5,2
Stripper temperature	XMEAS (18)	−20,1	5,3
Product separator temperature	XMEAS (11)	−19,5	5,9
Separator pot liquid flow (stream 10)	XMV (7)	−18,3	7,1
E feed flow (stream 3)	XMV (2)	−17,1	8,3
Product separator underflow (stream 10)	XMEAS (14)	−14,2	11,2
Stripper level	XMEAS (15)	−9,6	15,8
Stripper liquid product flow (stream 11)	XMV (8)	−3,6	21,8
Compressor recycle valve	XMV (5)	0,0	25,4
Stripper steam valve	XMV (9)	0,0	25,4
Recycle flow (Stream 8)	XMEAS (5)	0,0	25,4
Reactor feed rate (stream 6)	XMEAS (6)	0,0	25,4
Reactor level	XMEAS (8)	0,0	25,4
Stripper underflow (stream 11)	XMEAS (17)	0,0	25,4
D feed flow (Stream 2)	XMV (1)	0,0	25,4
D feed (stream 2)	XMEAS (2)	0,0	25,4
Purge valve (stream 9)	XMV (6)	29,1	54,5
Purge rate (stream 9)	XMEAS (10)	29,1	54,5
A and C feed flow (stream 4)	XMV (4)	32,7	58,0
A and C feed (stream 4)	XMEAS (4)	32,7	58,0
A feed (stream 1)	XMEAS (1)	32,7	58,0
A feed flow (stream 1)	XMV (3)	32,7	58,1
Product separator level	XMEAS (12)	34,8	60,2

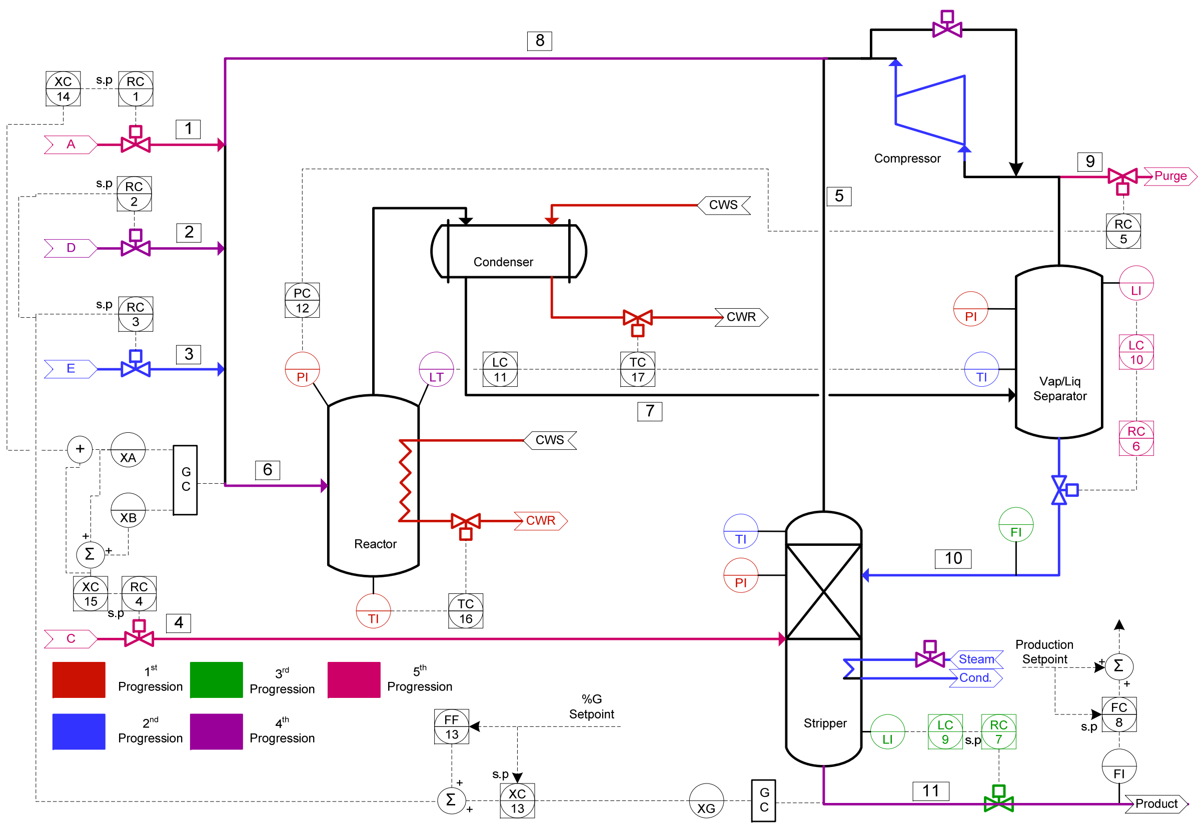


Figure 6.8: Flow diagram of the Tennessee Eastman Plant-wide Industrial Control Problem indicating causality.

- **Product separator pressure**: This signal (24, XMEAS(13)) is statistically correlated to signals 18 (reactor pressure, XMEAS(7)) and 20 (reactor temperature, XMEAS(9)). The equivalent correlation values are 0,91 and 0,58 respectively. These are highly correlated signals and for this reason, the MSTDE algorithm groups them together, in the region of the sticky reactor valve.

- **Stripper pressure**: This signal is statistically correlated to signals 18 (reactor pressure, XMEAS(7)), 20 (reactor temperature, XMEAS(9)) and 24 (product separator pressure, XMEAS(13)). The equivalent correlation values are 0,91, 0,58 and 0,92 respectively. We see a similar result as in the first case, and for this reason this signal is also grouped in the region of the sticky valve.

- **Condenser cooling water flow**: This signal is statistically correlated to signals 10 (reactor cooling water flow, XMV(10)), 20 (reactor temperature, XMEAS(9)) and 32 (reactor cooling water outlet temperature, XMEAS(21)). The equivalent correlation values are 0,52, 0,62 and −0,61 respectively. These values are not as large as the other cases but large in comparison with the rest of the signal correlations which explains the observed result.

It is up to the user to decide whether or not the “Product separator pressure”, “Stripper pressure” and the “Condenser cooling water flow” signals are of any relevance to the problem at hand. As with all the techniques used in this project, the result is not supposed to be taken as the absolute truth, but should rather assist the user in making a decision regarding the problem at hand.

Most of the measurements are under regulatory control, which influences the causal structure, and could yield counter-intuitive results. In addition, the original zeros (in the “Shift” column of table 6.5) could indicate be separate source signals as in the case in the synthetic data. For these reasons extreme caution should be taken when these results are interpreted.

All the zeros in the original shift column are as result of a statistically insignificant contribution towards the explained lags. This can be seen in figure 6.7(b), where signals 1 (D feed flow, XMV(1)), 13 (D feed stream, XMEAS(2)), 16 (Recycle flow, XMEAS(5)), 17 (reactor feed rate, XMEAS(6)), 19 (reactor level, XMEAS(8)), 28 (stripper underflow, XMEAS(17)) are neglected in the weighting function.

Signals 5 (compressor recycle valve, XMV(5)) and 9 (Stripper steam valve, XMV(9)) are zero for a different reason: they are correlated only to one another. For this reason they are grouped together and have no influence on the rest of the system.

CHAPTER 7

Results: Time Shifted Clustering

We combine the results of the previous two chapters. The data is shifted by the optimal calculated shift and then clustered by the K-means and K-medoid clustering algorithms. The optimal number of cluster centers for each data set is calculated as before.

7.1 Synthetic data

7.1.1 Optimal number of clusters

As in sections 5.1.1 and 5.2.1, the number of cluster centers, c, were increased from 2 to 15 in a step wise fashion. Figures 7.1(a) and 7.1(b) show the results for the validation indices when the K-means and K-medoid algorithms are applied to the optimally shifted synthetic data set. Note that these values are normalised with respect to the maximum value in the vector.

The validity indices follow the same trend as in sections 5.1.1 and 5.2.1. In both figures 7.1(a) and 7.1(b), the SC and S indices decrease after the initial spike in the values. There are local minima at $c = 4$ when using the K-medoid algorithm. This would suggest that there is some form of substructure in the data when $c = 4$. The profiles for both the SC and S indices are similar. The SC index is less sensitive to cluster size variations than the S index (refer to section 3.3). This suggests that the cluster sizes stay relatively equal as the number of cluster centers increase. The shapes of figures figures 7.1(a) and 7.1(b) suggest no optimal number of cluster centers, the same as in section 5.1.1.

Figure 7.2 shows the knee plots for both the K-means and K-medoid algorithms.

Both the cost functions indicate that the optimal number of clusters for the synthetic data set is $c = 3$, where the first knee appears. This result is comparable with that found

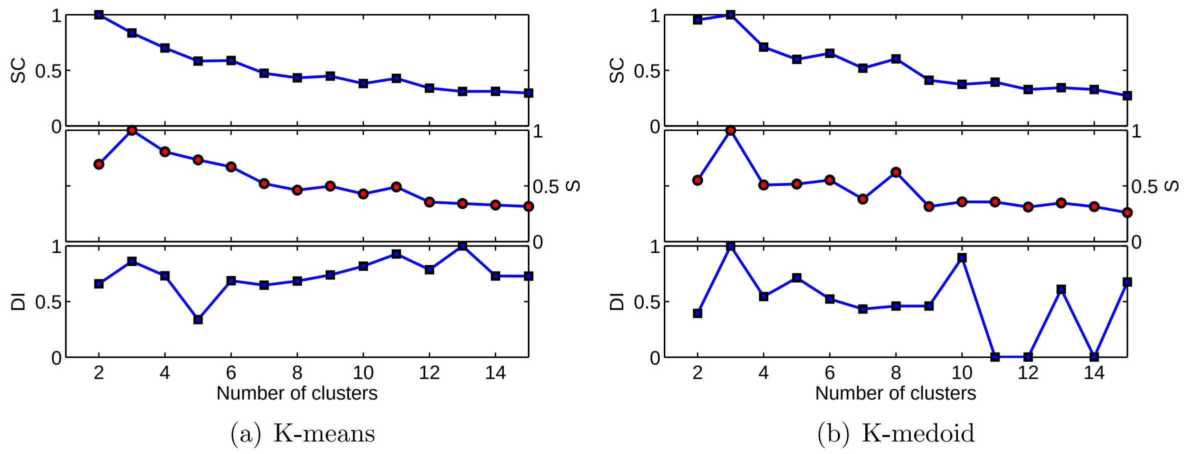


Figure 7.1: Validation indices for the optimal number of clusters using the K-means and K-medoid algorithms applied to the optimally shifted synthetic data set. The values are normalised with respect to the maximum value.

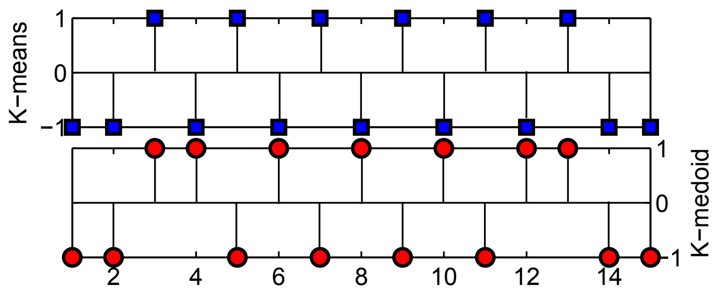


Figure 7.2: Second derivative knee plot from the cost functions of the respective clustering algorithms while using the optimally time shifted synthetic data.

in section 5.1.1, where the number of cluster centers was the same, $c = 3$. The DI for both the non-shifted and shifted data sets is at, or near, its maximum at the optimal number of clusters as determined with the knee plot (refer to figures 5.1 and 7.5).

7.1.2 Results with a fixed number of clusters

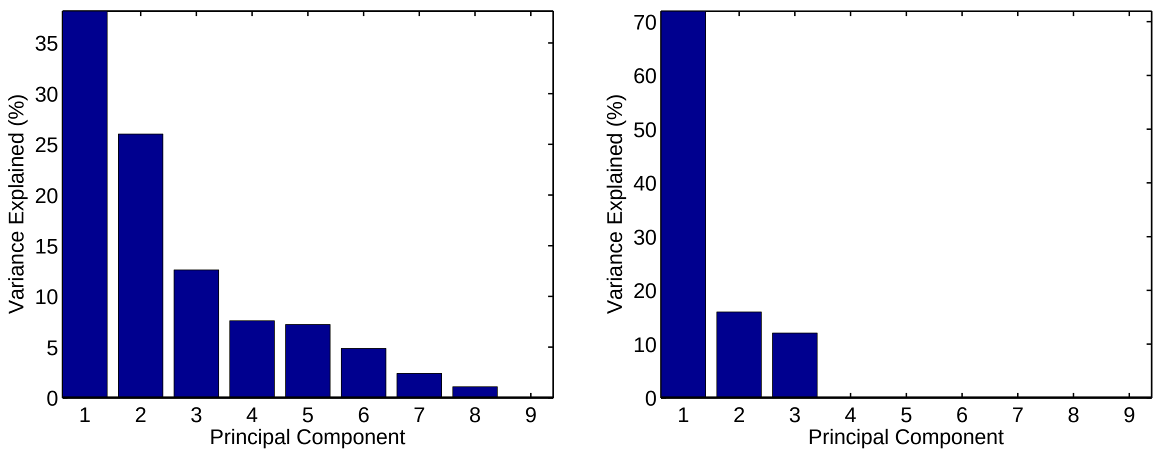
Validity Indices

The validity indices for the K-means and K-medoid algorithms are shown in table 7.1. When we compare these values to those obtained in section 5.1.2, where the synthetic data set was not shifted, we see that the values for both SC and S are smaller while that of the DI is slightly larger – for both the clustering algorithms.

Table 7.1: Validity indices for various clustering algorithms while using the time shifted synthetic data.

	K-means	K-medoid
SC	0,163	0,189
S	0,000 29	0,000 33
DI	0,098 2	0,137 9

This would suggest that the time shifting imparts more structure to the data set. The time shifting results in a data set that approaches the "best case" data set described in section 5.1.2. This is also evident when we consider the principal components. Figure 7.3 shows the variance explained by the principal components for the non-shifted and shifted data sets respectively.



(a) No time shift. (b) Time shifted data.

Figure 7.3: PCA: Variance explained by the principal components (synthetic data set).

The synthetic data set consists of 9 signals, divided into 3 groups of 3 signals. After the MSTDE algorithm has shifted the data set, we find that the 3 signals within each group align perfectly. Although there are 9 signals, the algorithm effectively reduced the number of signals to 3, of which the rest are just copies. This is evident from figure 7.3, where we find that all the signals contribute to the explained variance for the non-shifted data set, while only the first 3 contribute to the variance in the time-shifted data set. This can also be used to evaluate the effectiveness of the MSTDE algorithm. The more the variance is moved to the lower dimensional components, the better the signals are aligned in time.

Visual Clustering

In table 7.2, the dimensional reduction performance values for the respective clustering algorithms and dimensional reduction techniques are shown.

Table 7.2: Performance indicators for various clustering and dimensional reduction algorithms using the optimal time shifted synthetic data set.

Clustering	Dim. Red.	$p = \|U - U^*\|$	$\sum_{k=1}^{N} \overline{\mu_k^2}$	$\sum_{k=1}^{N} \overline{\mu_k^{2*}}$
K-means	PCA	0,140 1	1	0,682 2
	Sammon	0,140 9	1	0,681 1
	fuzzy-Sammon	0,240 8	1	0,603 1
K-medoid	PCA	0,212 2	1	0,599 4
	Sammon	0,213 3	1	0,605 3
	fuzzy-Sammon	0,263 5	1	0,589 4

The PCA technique performs the best for both the clustering algorithms. The Sammon mapping's performance is very close to the PCA's and is evident in figure 7.4. Both of these techniques seem to distinguish between different regions in the reduced dimensional space. The fuzzy Sammon mapping technique is the worst performer of the three techniques. This result is consistent with those obtained with the non-shifted data (refer to section 5.1.2). The fuzzy Sammon mapping shares some characteristics with the fuzzy clustering techniques where only the distance between the data points and the cluster centers are considered to be important refer to section 3.4.3). In section 5.1, we found that the fuzzy clustering algorithms were not successful at identifying proper substructures in the data. This also applies to the fuzzy Sammon dimensional reduction technique. This result is intuitive as these algorithms (fuzzy clustering and fuzzy dimensional reduction) share the same premise.

The PCA and Sammon mapping results in figure 7.4 show the same structures. The individual clusters are reminiscent of Lissajous figures.

The time series representation of figure 7.4 are shown in figures 7.5(a) and 7.5(b). When we compare these results with those obtained in figures 5.5(a) and 5.5(b), we find

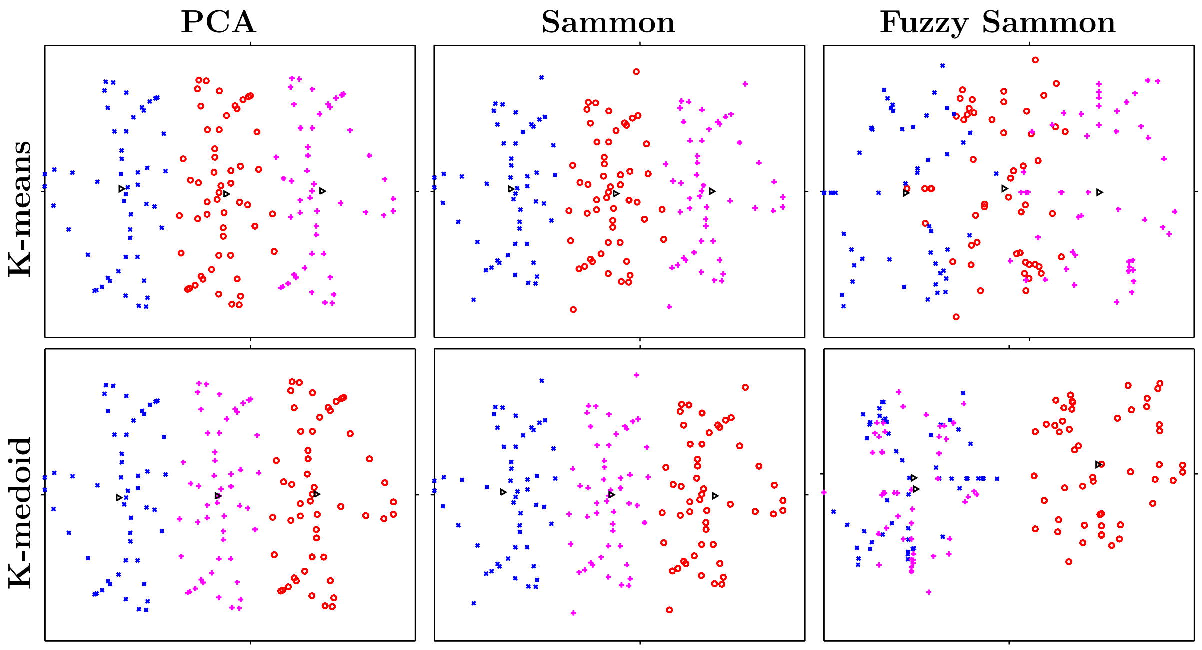
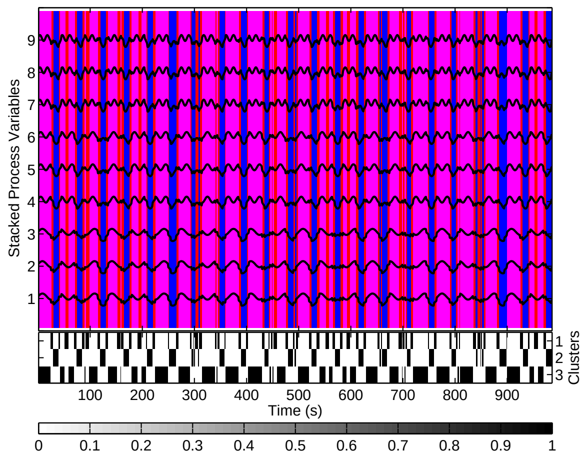


Figure 7.4: Visual representation of the K-means (top row) and K-medoid (bottom row) algorithms. The columns represent the different dimensional reduction techniques, starting with PCA (first column), Sammon mappings (second column) and fuzzy Sammon mappings (third column). All the axes are normalised from −1 to 1, with the tick on each axis indicating zero. Three clusters centers were used. These are indicated by the sideways triangles.

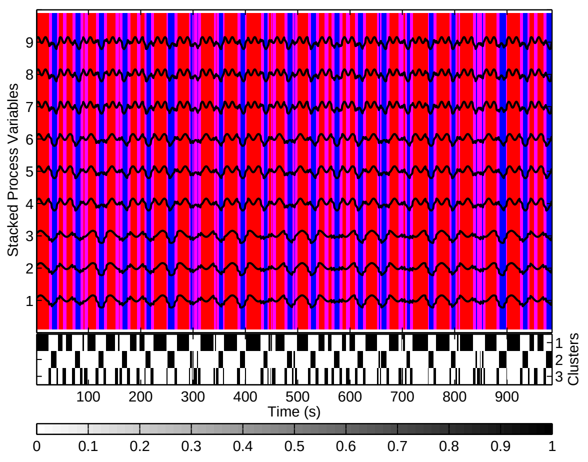
that the well defined cyclical behaviour has given way to a more sporadic time-cluster representation. However, cluster 3 and 1 for the K-means and K-medoid algorithms respectively, contain larger constituents than that of the non-shifted data set – where they were more uniform with respect to time. This is as result of the signal alignment due to the MSTDE algorithm. We do find different periods of repetition in the cluster sizes.

Initially, the number of data points in cluster 3 is large for the K-means algorithm. At $t \approx 50$ seconds, the number of data points in cluster 3 reduces but repeats at a higher frequency. This stops at $t \approx 200$ seconds, where the number of data points increase again. The amount of data points reduce in size again at $t \approx 550$ seconds an increase at $t \approx 720$. A similar result is obtained while using the K-medoid algorithm.

Therefore, the cyclical behavior has not disappeared completely, but the relative amount of data points in the 3 clusters have changed towards a single large grouping in cluster 3, a medium sized grouping in cluster 2 and a small grouping in cluster 1 (in time). What is interesting is that the sizes of the data point groupings in figure 7.4 appear to be the same.



(a) K-means



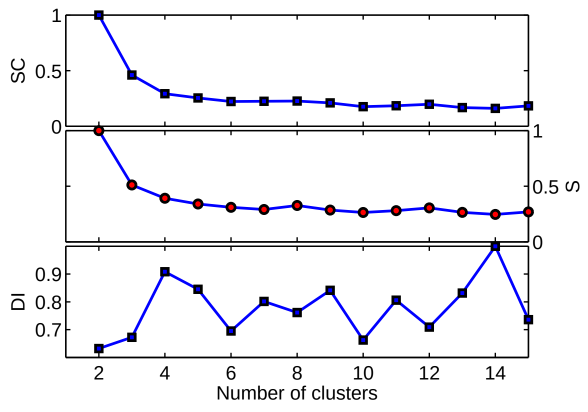
(b) K-medoid

Figure 7.5: Time representation of the different clustering algorithms for the optimally time shifted synthetic data set. They are shaded with different shades of gray to indicate to which cluster they belong at that point in time. The partial figure at the bottom indicates which cluster is "active" at that specific point in time.

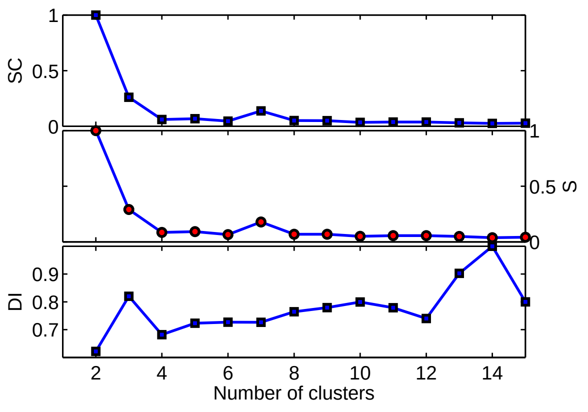
7.2 Tennessee Eastman Process

7.2.1 Optimal number of clusters

As in all the previous cases, the number of cluster centers were varied from 2 to 15 in a step wise fashion. Figures 7.6(a) and 7.6(b) show the resulting validation indices for the time-shifted Tennessee Eastman data set.



(a) K-means



(b) K-medoid

Figure 7.6: Validation indices for the optimal number of clusters using the K-means and K-medoid algorithms applied to the optimally shifted synthetic data set. The values are normalised with respect to the maximum value.

Figure 7.6(a) shows a gradual decline in the SC and S indices as the number of clusters are increased. The magnitudes of the normalised indices do not decrease as much as those of the corresponding K-medoid algorithm (shown in figure 7.6(b)). This suggests that the K-medoid algorithm finds much better substructures within the data from a much smaller number of cluster centers. The DI shows sporadic movements right through the cluster range for both the K-means and K-medoid algorithms as it cannot find structure within the data. Both the SC and S indices for the K-medoid algorithm would suggest that the optimal number of cluster centers should be in the region of $c = 3$, as there is a sudden decline after $c = 2$, after which it stays relatively constant. No conclusion can be drawn from the K-means results.

The resultant knee plot is shown in figure 7.7. The optimal number of clusters for the K-means algorithm occurs at $c = 4$, which differs form the non-shifted case (refer to section 5.2.2). This result suggests that the data substructure is more complex after the time shift than before the shift. This also becomes apparent in figures 7.10(a) and 7.10(b), where the resulting time cluster plots show a much more disjoint set of clusters. The optimal number of clusters for the K-medoid algorithm is at $c = 3$. This result is confirmed by the K-medoid validity indices.

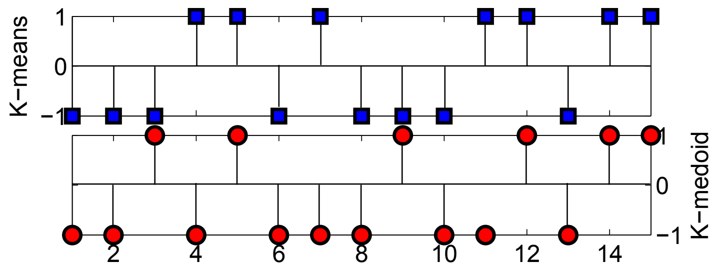


Figure 7.7: Second derivative knee plot from the cost functions of the respective clustering algorithms using the optimally time shifted Tennessee Eastman data.

7.2.2 Results with a fixed number of clusters

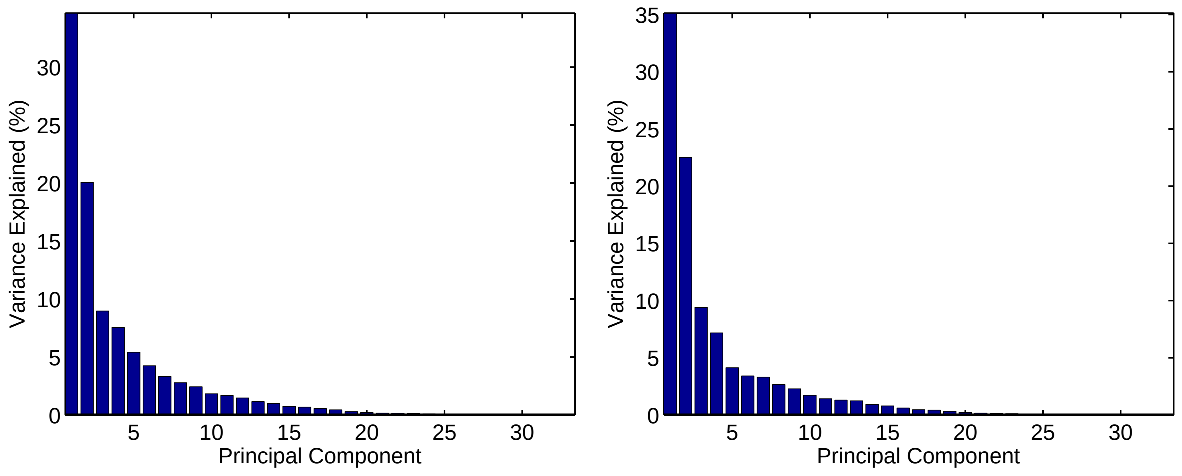
Validity Indices

Table 7.3 show the validity indices for the optimally time-shifted Tennessee Eastman data set.

Table 7.3: Validity indices for various clustering algorithms while using the time shifted Tennessee Eastman data.

	K-means	K-medoid
SC	0,285	0,637
S	0,000 5	0,000 95
DI	0,158	0,111

The SC and S indices are larger when these results are compared to than those obtained by the non-shifted data set (refer to section 5.2.2). This is an unexpected result, as the time shifted data should have a better defined substructure. However, the PCA analysis (figure 7.8) shows that the MSTDE algorithm has shifted the variance to the lower dimensions as was the case for the synthetic data.



(a) No time shift. (b) Time shifted data.

Figure 7.8: PCA: Variance explained by the principal components (Tennessee Eastman data set).

Visual Clustering

In table 7.4, the dimensional reduction performance values for the respective clustering algorithms and dimensional reduction techniques are shown.

Table 7.4: Performance indicators for various clustering and dimensional reduction algorithms using the optimal time shifted Tennessee Eastman data set.

Clustering	Dim. Red.	$p = \|U - U^*\|$	$\sum_{k=1}^{N} \overline{\mu_k^2}$	$\sum_{k=1}^{N} \overline{\mu_k^{2*}}$
K-means	PCA	0,195 1	1	0,536 0
	Sammon	0,354 3	1	0,345 1
	fuzzy-Sammon	0,303 8	1	0,455 4
K-medoid	PCA	0,308 4	1	0,383 1
	Sammon	0,353 9	1	0,384 9
	fuzzy-Sammon	0,346 1	1	0,369 0

As with the synthetic data set, we find that the PCA dimensional reduction technique performs best. However, the fuzzy-Sammon mapping performs marginally better than the Sammon mapping. The overall performance is worse than that of the non-shifted data (refer to section 5.2.2). This result aligns with that of the previous section.

The visual representation of the clustering results are shown in figure 7.9. It is clear that large amounts of overlapping occur for both the K-means and K-medoid algorithms. This result is also confirmed by the validity indices in table 7.3.

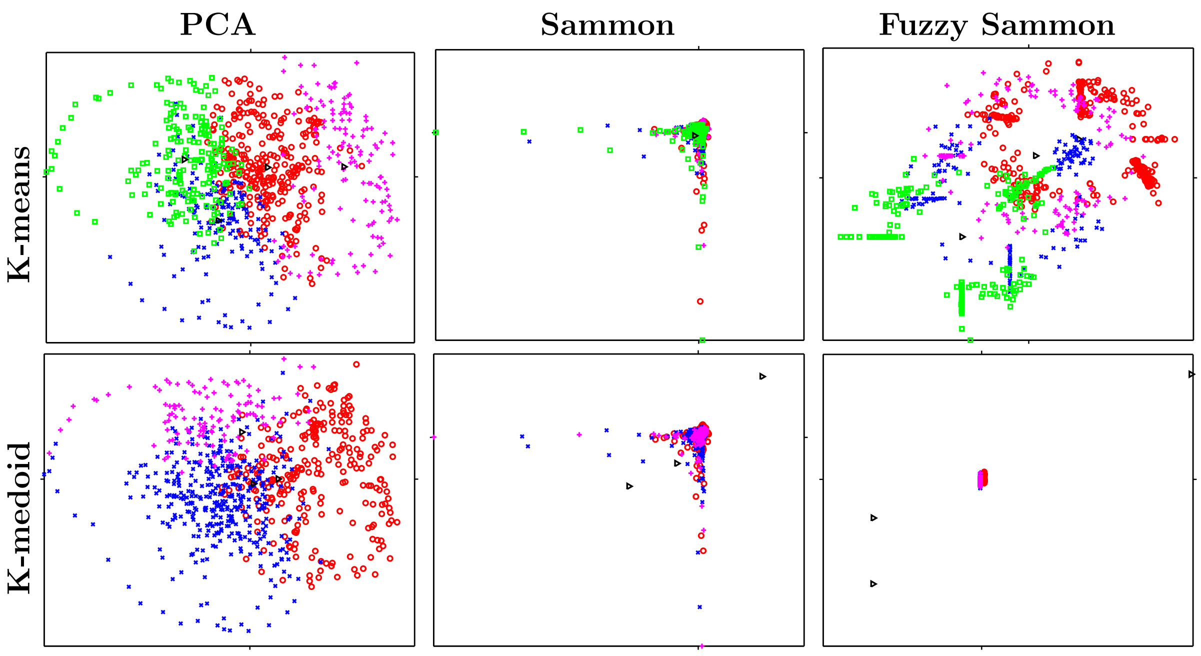


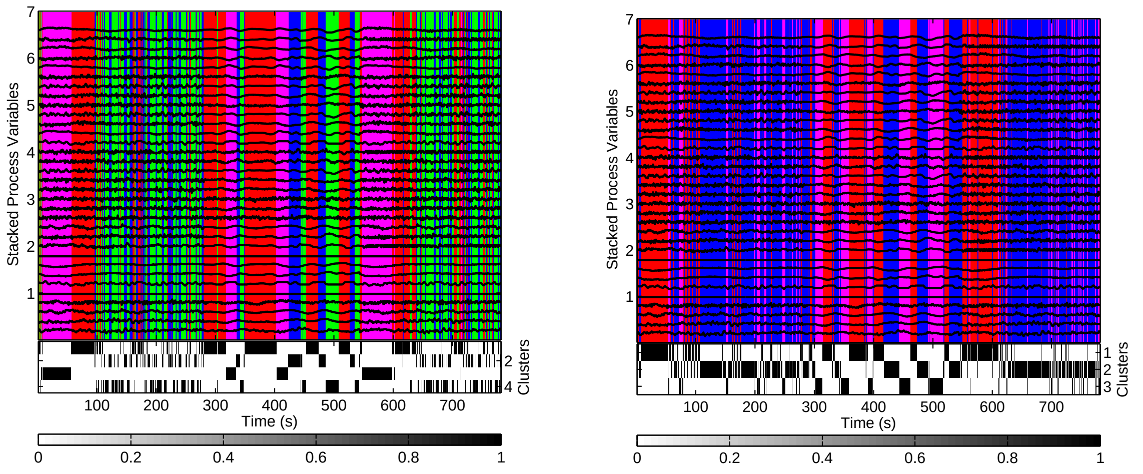
Figure 7.9: Visual representation of the K-means (top row) and K-medoid (bottom row) algorithms on the optimally time shifted Tennessee Eastman data. The columns represent the different dimensional reduction techniques, starting with PCA (first column), Sammon mappings (second column) and fuzzy Sammon mappings (third column). All the axes are normalised from -1 to 1, with the tick on each axis indicating zero. Three cluster and four centers were used for the K-means and K-medoid algorithms respectively. These are indicated by the sideways triangles.

The Sammon mapping is not able to separate the clusters in either instance. This appears to be the case for the K-medoid fuzzy Sammon mapping as well, but upon closer inspection, we find that the dimensionally reduced data lie extremely close to one another. There is however a marginal distinction between the different data even though they are far away from their respective cluster centers.

Figures 7.10(a) and 7.10(b) show the time-cluster results for both the K-means and K-medoid clustering algorithms respectively.

The K-means algorithm does not succeed in identifying the three operating regions as was the case for the non-shifted data (figures 5.10(a) and 5.10(b)). One of the possible causes could be due to the increased number of cluster centers. However, it is felt that due to the fact that the MSTDE algorithm introduced a higher measure of correlation back into the system – by aligning the signals in time – causing the clustering algorithm to identify much smaller regions with similar characteristics. This results in a large number of cluster stripes (and excessive striping).

The K-medoid algorithm produces a similar looking time-cluster plot to that of the non-shifted data however, table 7.3 suggests that this is not the case.



(a) K-means.

(b) K-medoid.

Figure 7.10: Time representation of the different clustering algorithms for the optimally time shifted Tennessee Eastman data set. The solid lines on the main plot represent the process signals. They are shaded with different shades of gray to indicate to which cluster they belong at that point in time. The partial figure at the bottom indicates which cluster is "active" at that specific point in time.

7.3 Introduction of a Time Vector

In all the previous sections, the time vector of the data was not clustered together with the data. This makes intuitive sense as it does not contain relevant process data.

From all the time-cluster plots (figures 5.5(a) to 5.5(d), 5.10(a) and 5.10(b), 7.5(a) and 7.5(b), 7.10(a) and 7.10(b)), it is evident that a degree of striping occurs. Due to our prior knowledge of the Tennessee Eastman data set, we would expect the clustering algorithm to identify 3 separate data sets, as they differ with respect to the disturbance acting in each of them.

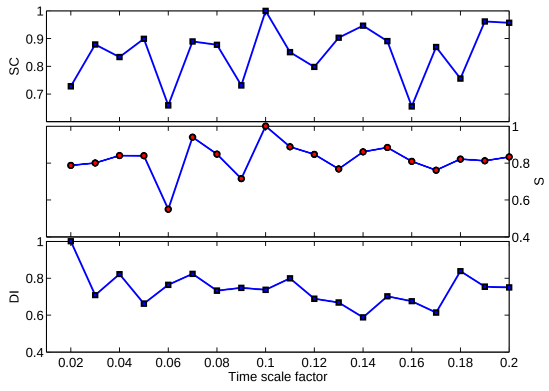
The data matrix is altered to include the associated time vector. This is shown in equation 7.1. The original data set is mean centred and scaled to unit standard deviation. However, the time vector is not mean centred and scaled to unit variance due to the nature of the vector. To have some form of control with respect to weight it has in the clustering algorithm, a scale factor, α, is introduced. This is done to decrease the weight the original time vector has on the clustering algorithm, due to its inherent large values.

$$X = \begin{pmatrix} \zeta t_1 & x_{11} & x_{12} & \dots & x_{1n} \\ \zeta t_2 & x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \zeta t_N & x_{N1} & x_{N2} & \dots & x_{Nn} \end{pmatrix} \tag{7.1}$$

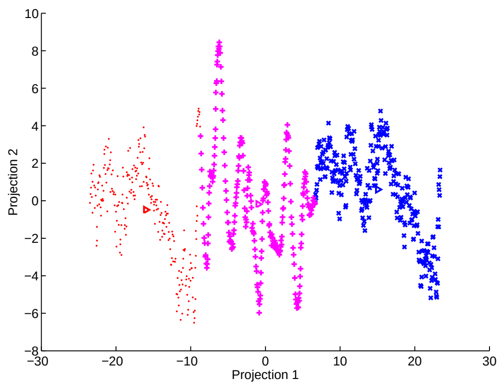
Figure 7.11 shows the result when a scaled time vector is added to the clustering process. In an attempt to obtain the optimal scaling factor, ζ was increased from $\zeta = 0{,}02$ to

$\zeta = 0{,}2$ in steps of 0,01. These bounds were determined visually, as the bottom bound resulted in the same clustering without the time hyperplane and the top bound resulted in a distorted view away from reality. Figure 7.11(a) shows this result. Both the SC and S indices show a minimum at $\zeta = 0{,}06$, while the DI is sporadic right through the range.

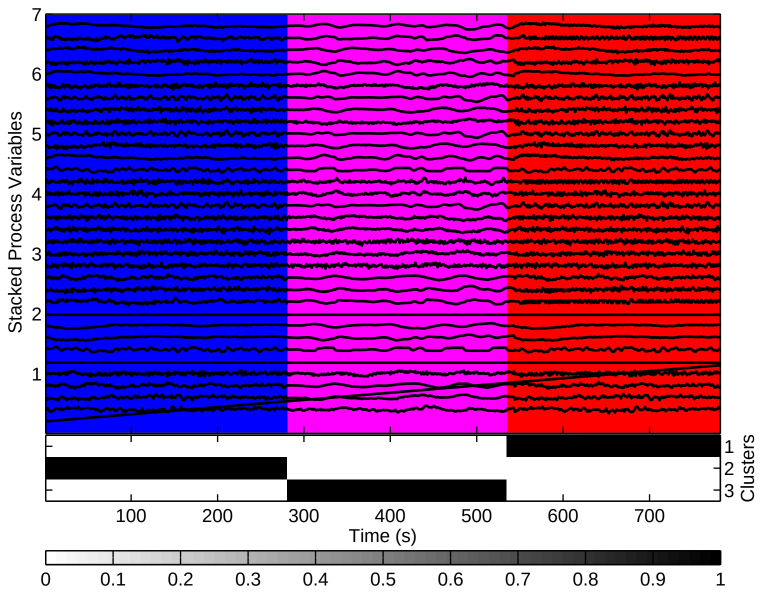
Figure 7.11(b) shows the principal component analysis representation of the data when $\zeta = 0{,}06$. When this is compared to the equivalent Tennessee Eastman representation without the time hyperplane (refer to figure 7.9), we find a much clearer separation of clusters. This is also confirmed by figure 7.11(c), where the 3 clusters represent the 3 different operating regions.



(a) Validity indices.



(b) Principal component visualisation.



(c) Time cluster plot.

Figure 7.11: Addition of a time vector to the clustered data (time shifted Tennessee Eastman data set). The scale factor is $\zeta = 0{,}06$.

This result would suggest that the addition of the time hyperplane reduces the effect of striping (as the impetus is shifted towards the clustering of the time vector itself).

This is due to the time vector relating data points that are have similar spatial features that are close to each other in time. However, further work is needed to give meaningful conclusions.

CHAPTER 8

Conclusions, Recommendations and Future Work

The ability of the following techniques were investigated on a synthetic data set as well as modified Tennessee Eastman data:

Multiple Signal Time Delay Estimation (MSTDE) The use of the cross-correlation function (CCF), combined with statistical thresholding techniques were used to calculate optimal time shifts for causally linked signals.

Clustering Algorithms The K-means, K-medoid, fuzzy C-means, Gath-Geva and Gustafson-Kessel were used to cluster these data sets

Dimensional Reduction Techniques The principal component analysis (PCA), Sammon mapping and fuzzy-Sammon mapping techniques were used to enable interpretation of high dimensional clustered data.

8.1 Multiple Signal Time Delay Estimation (MSTDE)

The MSTDE algorithm proved to be a simple yet powerful solution for time delay estimation problems concerned with multiple signals. Because it is posed as a linear problem,

- It is numerically and computationally efficient.
- The pseudo-inverse supplies an easy generalisation for the least-squares solution of non-square inversions.

When a data set contains many signals, some of them will not be correlated to each other. Bauer & Thornhill (2008) provided a powerful method to determine whether these signals are statistically correlated or not. This, combined with the correlation coefficients R, provided an excellent weighting function. It also proved to solve large signal sets the best by having the smallest residual norm when compared to no weighting and using only the correlation coefficients.

8.2 Clustering Algorithms

Five clustering algorithms were initially chosen: the K-means, K-medoid, fuzzy C-means, Gath-Geva and Gustafson-Kessel algorithms. The Gath-Geva algorithm was never used as a numerically stable solution could never be developed.

To determine the performance of these algorithms, the following cluster validation metrics were used: the Partition Coefficient (PC), Classification Entropy (CE), Partition Index (SC), Separation Index (S), Dunn's Index (DI) and the alternative Dunn Index (ADI).

These algorithms were evaluated against two base case data sets to evaluate the clustering abilities of the clustering algorithms and to evaluate the expected ranges of the performance metrics.

- A best case scenario which included 3 signals and 3 well defined clusters. All the validation indices except the PE and CE indices yielded intuitive results, being either 0 or inf. The K-means, K-medoid and fuzzy C-means algorithms proved effective in determining the 3 clusters. The Gustafson-Kessel algorithm also identified 3 clusters, but these were not well defined and resulted no useful information.

- A worst case scenario which included a time series set of 9 random signals with the same mean and variances as the synthetic data set defined in section 2.1. All the validation indices yielded values far from the ideal case.

These validity indices showed little promise in identifying the optimal number of clusters. The optimum number of clusters were found by making use of a "knee" plot (analogues to the second derivative) of the cost function of each clustering algorithm. This approach yielded consistent results for both the non-shifted and shifted data sets.

The performance of the clustering algorithms varied for the non-shifted and shifted data sets.

8.2.1 Non-Shifted

This data set contained all the original data. The synthetic data set yielded good results for all the clustering algorithms except the Gustafson-Kessel algorithm. The validity indices for the GK algorithm resembled those of the random value data set, which indicates that this clustering algorithm finds no substructure within the data. The remaining algorithms produced a consistent repeating pattern when the time-series clustering plots are observed. This suggests that the algorithms find repeating structures in the data.

The influence of noise was also investigated when using the clustering algorithms. The results indicated that the K-means algorithm's performance decreased with an increase in the noise power, up to a noise power of 1×10^{-1}, after which the indices show a increase in performance. The K-medoid showed a continual performance improvement throughout the noise range. Both of these results are counter-intuitive and it seems that high noise powers impart a certain structure within the data.

The number of clustering algorithms was reduced from the original 5 to 2. The K-means and the K-medoid algorithms. None of the fuzzy algorithms seem to yield informative results when using complex data. The number of validity indices was also reduced to 3: the SC, S and DI metrics. The PC and CE indices only produce monotonically increasing or decreasing values as the number of cluster centers increase. The ADI did not yield any useful information.

The K-means algorithm provided good results when clustering the Tennessee Eastman process data. It was able to distinguish between the three different modes of operation (evident from the time series cluster plot). This would give the operator insight into when the plant operated in different regions. However, the K-medoid algorithm was less successful in identifying the different operating regions. It produced sporadic cluster time clusters which would not assist the plant operator in identifying different modes of plant operation.

8.2.2 Shifted

This data set was shifted by making use of the MSTDE algorithm. Consequently, it was marginally smaller than the original data set due to the shift and trimming of the edges.

The synthetic data set showed comparable results to those obtained without the time shift. The validity indices were marginally better, suggesting that the optimal time shift imparts more structure to the data set. This was confirmed when comparing the principal components of the non-shifted and shifted data sets. The non-shifted data have a greater variation among the data, hence a larger number of principal components. The shifting reduced the effective number of signals from 9 to 3, therefore reducing the number of principal components. Instead of clusters with more or less the same size, the effect of the MSTDE algorithm increased the size of one of the clusters, while keeping one approximately the same and reducing the last cluster size. The sequence of the clusters in the patterns changed, but this does not have any substantial meaning.

The results for the time shifted Tennessee Eastman differed substantially from the non-shifted data set. The optimum number of cluster centers were $c = 4$ for the K-means algorithm compared to the $c = 3$ for the K-medoid clustering algorithm. The K-means time series clustering plots indicate excessive striping when compared to the non-shifted data set. This suggests that the MSTDE algorithm reduces the structure within the data. The K-medoid algorithm's result compares with that obtained in the non-shifted case.

The explicit addition of a time vector in the data yields promising result. The premise of this method is in the fact that it weights data points that are closer to each other to be clustered together. This would eliminate the excessive striping that is observed in many of the time series cluster plots. However, this technique does require more work.

8.3 Dimensional Reduction Techniques

Principal component analysis (PCA) performed the best of all the dimensional reduction techniques. Feil et al. (2007) proposed that the fuzzy-Sammon mapping performs better than the Sammon mapping, but our results differed. In most instances the Sammon mapping performed better than the fuzzy-Sammon mapping.

The fuzzy-Sammon mapping makes use of a fuzzy membership function, E_{fuzz}, similar to those used in the fuzzy clustering algorithms. It is clear that all the fuzzy clustering techniques did not perform very well, as it struggled to identify structure within the data. This seems to be the downfall of the fuzzy-Sammon mapping as well. No clear structure was apparent in the dimensional reduction plots.

8.4 Future Work

Clustering The effect of time on the clustering algorithm needs to be further investigated. Either the time needs to be implicitly built into the data set or the algorithm needs to weight data points that are closer in time to reduce the amount of striping when a time series cluster plot is observed.

Time Delay Estimation and Causality Bauer & Thornhill (2008) uses the CCF function as well as the statistical thresholding functions to infer causality from the resultant time delays. However, their approach is complex and requires matrix pivoting to determine the sequence of events. Our MSTDE algorithm shows promise in terms of inferring causality from the resultant time shifts. The linear algebra approach resolves all the matrix pivoting issues in their method by using pre-optimised algorithms to handle all the pivoting.

Sliding Time Window It is likely that more than one disturbance will occur in a data set. As shown in chapter 6, the calculated time delays are not accurate if more than one disturbance occurs in a data set. Therefore, a sliding window could be used to continually monitor the correlation matrix as time passes. If the matrix changes sufficiently, another disturbance may have entered the system. This would aid online assistive technologies and TDE algorithms.

BIBLIOGRAPHY

Abonyi, J.; Migaly, S. and Szeifert, F. (2002) "A hybridized approach to data clustering", *Advances in Soft Computing, Engineering Design and Manufacturing,* pages 99–108.

Abonyi, J.; Feil, B.; Nemeth, S. and Arva, P. (2005) "Modified GathGeva clustering for fuzzy segmentation of multivariate time-series", *Fuzzy Sets and Systems,* (149), 3956.

Ash, J. N. and Moses, R. L. (2005) "Acoustic time delay estimation and sensor network self-localization: Experimental results", *J. Acoust. Soc. Am., 118* (2), 841–850.

Balasko, B.; Abonyi, J. and Feil, B. "Fuzzy Clustering and Data Analysis Toolbox", Matlab Toolbox (2005).

Barkat, M. (1991) *Signal Detection & Estimation*, Artech House, London.

Bauer, M. (2005) *Data-Driven Methods for Process Analysis*, PhD thesis, University College London, London.

Bauer, M. and Thornhill, N. (2008) "A practical method for identifying the propagation path of plant-wide disturbances", *Journal of process control, 18*, 707–719.

Bensaid, A.; Hall, L.; Bezdek, J.; Clarke, L.; Silbiger, M.; Arrington, J. and Murtagh, R. (1996) "Validity-Guided (Re)Clustering with applications to Image Segmentation", *IEEE Transactions on Fuzzy Systems, 4* (2), 112–123.

Bezdek, J. (1981) *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York.

Bezdek, J. and Pal, R. (1995) "An index of topological preservation for feature extraction", *Pattern Recognition, 28* (3), 381–391.

da Silva, S.; Junior, M.; V.L., J. and Brennan, M. (2008) "Structural damage detection by fuzzy clustering", *Mechanical Systems and signal Processing, 22*, 1636–1649.

Downs, J. and Vogel, E. (1993) "A plant-wide industrial process control problem", *Computers chem. Engng., 17* (3), 245–255.

Feil, B.; Balasko, B. and Abonyi, J. (2007) "Visualization of fuzzy clusters by fuzzy Sammon mapping projection: Application to the analysis of phase space trajectories", *Soft Comput.,* (11), 479–488.

Filippone, M.; Camastra, F.; Masulli, F. and Rovetta, S. (2008) "A survey of kernel and spectral methods for clustering", *Pattern Recognition, 41*, 176–190.

Franc, V. and Hlavac, V. "Statistical Pattern Recognition Toolbox for Matlab: Users guide", (2004) URL `ftp://cmp.felk.cvut.cz/pub/cmp/articles/franc/Franc-TR-2004-08.pdf`.

Frank, P. (1990) "Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy- A Survey and Some New Results", *Automatica, 26* (3), 459–474.

Gray, R. M. and Neuhoff, D. L. (1998) "Quantization", *IEEE Trans Inform Theory, 44*, 2325–2385.

Güngör, Z. and Ünler, A. (2008) "K-Harmonic means data clustering with tabu-search method", *Applied Mathematical Modelling, 32*, 1115–1125.

Han, J. and Kamber, M. (2006) *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, Amsterdam.

Kao, Y.-T.; Zahara, E. and Kao, I.-W. (2008) "A hybridized approach to data clustering", *Expert Systems with Applications, 34*, 1754–1762.

Keogh, E.; Lin, J. and Truppel, W. "Clustering of Time Series Subsequences is Meaningless: Implications for Previous and Future Research", Proceedings of the Third IEEE International Conference on Data Mining (2003).

Knapp, C. and Carter, G. (1976) "The generalized correlation method for estimation of time delay", *Acoustics, Speech and Signal Processing, IEEE Transactions on, 24* (4), 320–327.

Lerner, B.; Guterman, H.; Aladjem, M.; Dinstein, I. and Romem, Y. (1998) "On pattern clasification with Sammon's nonlinear mapping, an experimental study", *Pattern Recognition, 31* (4), 371–381.

Liu, J. and Xu, M. (2008) "Kernalized fuzzy attribute C-means clustering algorithm", *Fuzzy Sets and Systems, 159*, 2428–2445.

Liu, T.; Ye, Y.; Zeng, X. and Ghannouchi, F. (2008) "Accurate Time-Delay Estimation and Alignment for RF Power Amplifier/Transmitter Characterization", *Circuits and Systems for Communications, 2008. ICCSC 2008. 4th IEEE International Conference on,* pages 70–74.

MacKay, D. (2007) *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, Cambridge.

Martin, E. B. and Morris, A. J. (1996) "Non-parametric confidence bounds for process performance monitoring charts", *J. Proc. Cont., 6* (6), 349–358.

Maszczyk, T. and Duch, W. "Support Vector Machines for visualization and dimensionality reduction", (2008) URL `http://www.fizyka.umk.pl/publications/kmk/08-SVM-vis.pdf`.

Merz, C. and Murphy, P. "UCI repository of machine learning databases (1998-2004)", (2008) URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Pal, N. and Bezdek, J. (1995) "On cluster Validity for the Fuzzy c-Means Model", *IEEE Transactions on Fuzzy Systems, 3* (3), 370–379.

Park, H. and Jun, C. (2008) "A simple and fast algorithm for K-medoids clustering", *Expert Systems with Applications, In press*, doi:10.1016/j.eswa.208.01.039.

Phillpotts, D. (2007) "Nonlinear Fault Detection and Diagnosis using Kernel based Techniques applied to a Pilot Distillation Column", Master's thesis, University of Pretoria, Pretoria, South Africa.

Richard, A. and Dean, W. (2007) *Applied Multivariate Statistical Analysis*, Pearson - Prentice Hall, London.

Ricker, N. (1996) "Decentralized control of the Tennessee Eastman Challenge Process", *Journal of Process Control, 6* (4), 205–221.

Ricker, N. "Tennessee Eastman Challenge Archive", (2008) URL `http://depts.washington.edu/control/LARRY/TE/download.html`.

Sammon, J. (1969) "A Nonlinear Mapping for Data Structure Analysis", *IEEE Transactions on Computers, C18* (5), 401–409.

Singhal, A. and Seborg, D. (2005) "Clustering Multivariate Time-Series Data", *Journal of Chemometrics, 19*, 427–438.

van der Maaten, L.; Postma, E. and van den Herik, H. "Dimensionality Reduction: A comparative review", IEEE Transactions on Pattern Analysis and Machine Intelligence (submitted) (2007).

Venkatasubramanian, V.; Rengaswamy, R. and Yin, K. (2003) "A Review of Process Fault Detection and Diagnosis - Part 3: Process History Based Methods", *Computers and Chemical Engineering,* (27), 327346.

Wang, W. and Zhang, Y. (2007) "On fuzzy cluster validity indices", *Fuzzy Sets and Systems, 158*, 2095–2117.

Wu, X.; Kumar, V.; Quinlan, J. R.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G. J.; Ng, A.; Liu, B.; Yu, P. S.; Zhou, Z.-H.; Steinbach, M.; Hand, D. J. and Steinberg, D. (2008) "Top 10 algorithms in data mining", *Knowl. Inf. Syst., 14*, 1–37.

Xie, X. and Beni, G. (1991) "A Validity Measure for Fuzzy Clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence, 13* (8), 841–847.

Yu, Z.; Zhou, X. and Wu, Z. (2006) "Transient Stability Boundary Visualization for Power System", *International Conference on Power System Technology, 1* (1), Art. No. 4116189.