DIAGNOSTIC MONITORING OF DYNAMIC SYSTEMS USING ARTIFICIAL IMMUNE SYSTEMS

by

Charl Maree

Thesis submitted in partial fulfilment

of the requirements for the Degree

of

MASTER OF SCIENCE IN ENGINEERING

(CHEMICAL ENGINEERING)

In the Department of Process Engineering

at the University of Stellenbosch

Supervised by

Prof. C. Aldrich

STELLENBOSCH

DECEMBER 2006

DECLARATION

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Signature: …………………………

Date: ……………………………….

SUMMARY

The natural immune system is an exceptional pattern recognition system based on memory and learning that is capable of detecting both known and unknown pathogens. Artificial immune systems (AIS) employ some of the functionalities of the natural immune system in detecting change in dynamic process systems. The emerging field of artificial immune systems has enormous potential in the application of fault detection systems in process engineering.

This thesis aims to firstly familiarise the reader with the various current methods in the field of fault detection and identification. Secondly, the notion of artificial immune systems is to be introduced and explained. Finally, this thesis aims to investigate the performance of AIS on data gathered from simulated case studies both with and without noise.

Three different methods of generating detectors are used to monitor various different processes for anomalous events. These are:

- Random Generation of detectors
- Convex Hulls
- The Hypercube Vertex Approach

It is found that random generation provides a reasonable rate of detection, while convex hulls fail to achieve the required objectives. The hypercube vertex method achieved the highest detection rate and lowest false alarm rate in all case studies. The hypercube vertex method originates from this project and is the recommended method for use with all real valued systems, with a small number of variables at least.

It is found that, in some cases AIS are capable of perfect classification, where 100% of anomalous events are identified and no false alarms are generated. Noise has, expectedly so, some effect on the detection capability on all case studies. The computational cost of the various methods is compared, which concluded that the hypercube vertex method had a higher cost than other methods researched. This increased computational cost is however not exceeding reasonable confines therefore the hypercube vertex method nonetheless remains the chosen method.

The thesis concludes with considering AIS's performance in the comparative criteria for diagnostic methods. It is found that AIS compare well to current methods and that some of their limitations are indeed solved and their abilities surpassed in certain cases. Recommendations are made to future study in the field of AIS. Further the use of the Hypercube Vertex method is highly recommended in real valued scenarios such as Process Engineering.

OPSOMMING

Die natuurlike immuunstelsel is uitstekend aangepas om patrone te herken deur gebruik te maak van geheue en vorige ondervinding. Dit is in staat om beide bekende en onbekende patogene te identifiseer. Kunsmatige immuunstelsels spoor veranderinge in dinamiese prosesse op deur van sekere funksionaliteite van die natuurlike immuunstelsel gebruik te maak. Die groeiende veld van kunsmatige immuunstelsels het enorme potensiaal in die toepassing van fout opsporing in proses ingenieurswese.

Die doel van die tesis is nou om eerstens die leser met huidige fout deteksie metodes bekend te maak, asook die konsep van kunsmatige immuunstelsels voor te stel en te verduidelik. Uiteindelik is die tesis gemik om die reaksie van kunsmatige immuunstelsels op gesimuleerde gevallestudies (met en sonder geraas) te toets.

Drie verskillende metodes word gebruik om detektore te genereer en sodoende die proses te monitor vir abnormale gedrag. Hierdie metodes is:

- Willekansige detektore
- Konvekse Hulls
- Die “Hypercube Vertex” benadering

Willekansige detektore toon ‘n redelike vermoë om foute te identifiseer, waar konvekse hulls oneffektief blyk. Die “hypercube vertex” metode het in al die gevalle studies uitgeblink deur die hoogste deteksie en laagste persentasie vals alarms te genereer. Die “hypercube vertex” metode is in die projek geskep en word as die metode van keuse vir alle reële prosesse aangewys, of te wel prosesse met relatiewe lae dimensies .

In sommige gevalle was die kunsmatige immuunstelsel in staat tot perfekte klassifikasie, waar 100% van die abnormale gevalle geïdentifiseer is sonder dat enige vals alarms gegenereer is. Geraas het na verwagting wel ‘n invloed op die deteksie vermoë in al die gevallestudies gehad. Die berekeningskoste in elke gevallestudie is met mekaar vergelyk, wat vorendag gebring het dat die “hypercube vertex” metode die hoogste koste van al die metodes behels. Hierdie verhoogde berekenings intensiteit van die “hypercube vertex” metode oortref nie moderne rekenaar kapasiteite nie en word dus steeds as die metode van keuse voorgestel.

Hierdie tesis sluit af deur kunsmatige immuunstelsels se eienskappe in die lig van die vergelykbare kriteria vir diagnostiese metodes te ondersoek. Hieruit is bevind dat die kunsmatige immuunstelsels goed met huidige metodes vergelyk. Sommige tekortkominge van huidige metodes is oorskry en hulle vermoëns oortref. Voorstelle, wat vir toekomstige navorsers voordelig kan wees, word ook ten slotte bygevoeg. Eindelik word die “hypercube vertex” metode hoogs aanbeveel vir alle reël-getallige prosesse soos wat in die veld van proses ingenieurswese gevind word.

Having many things to write unto you, I would not write with paper and ink: but I trust to come unto you, and speak face to face, that our joy may be full.

-The Holy Bible: 2 John 12

ACKNOWLEDGEMENTS

My sincere thanks to the following people for hours of counselling, valuable advice and indispensable guidance:

Dad (Basie Maree), Mom (Joey Maree) and Prof. Chris Aldrich

…and to everyone in the lab for their willingness to put aside their work and share their attention to whatever it is I had to say, be it work related or just about anything else:

Yolandi Muntingh, Paul Botha, John Burchell, Jaco Nel, Rassie van der Westhuizen and Gorden Jemwa

Finally, I would like to express a special thanks to my closest friends and family who put up with me in the trying times, but especially for being there during the good times. Thanks for endless hours of laughter, wine and bad jokes.

Jacques Maree, Yolandi Muntingh, Jaco Nel, Paul Botha, Ettiene du Plessis, Lafras Moolman, and Pierre Fourie

Without the help of these people, Artificial Immune Systems would not have seen its way into this world through this thesis.

Nomenclature

ab	Antibody
AEM	Abnormal Event Management
ag	Antigen
AIS	Artificial Immune System
d	Euclidean Distance
D	Dimension
DT-AIS	Danger Theory Artificial Immune Systems
EKF	Extended Kalman Filter
L	Lag
M	Nonself space
m1	Drifting data
m2	Drifted data
N	Number of Detectors
PCA	Principal Component Analysis
PLS	Partial Least Squares
QTA	Qualitative Trend Analysis
R	Detector set
RL	Autocorrelation function
ROC	Received Operating Characteristic
S	Self space
T	Time series data
U	Hamming shape space
δ	Distance between detectors and corresponding self event
ε	Affinity threshold

The symbols defined above may briefly be used for meanings other than stipulated here. In this case and for miscellaneous symbols not defined here, the symbols are defined where applicable.

TABLE OF CONTENTS:

Chapter 1: Introduction

Process anomaly detection is becoming increasingly important in modern process facilities. Timely and accurate detection of deviations from normal process behaviour can prevent abnormal event progression and reduce the occurrence of process incidents. These incidents can and usually do, lead to accidents that cause injury, profit loss, environmental degradation, damage to equipment, or product quality decline. Industrial statistics show that roughly 70% of industrial accidents are caused by human errors. For example, the petrochemical industries in the USA alone lose an estimated 20 billion dollars annually (Nimmo, 1995) and they have rated abnormal event management (AEM) as their number one obstacle to overcome. The automation of process fault detection and identification forms the first step in AEM. Fault detection and identification is therefore of fundamental importance to the 21st century industries. Various institutions in South Africa research this field, some of which include SASOL and Anglo Platinum.

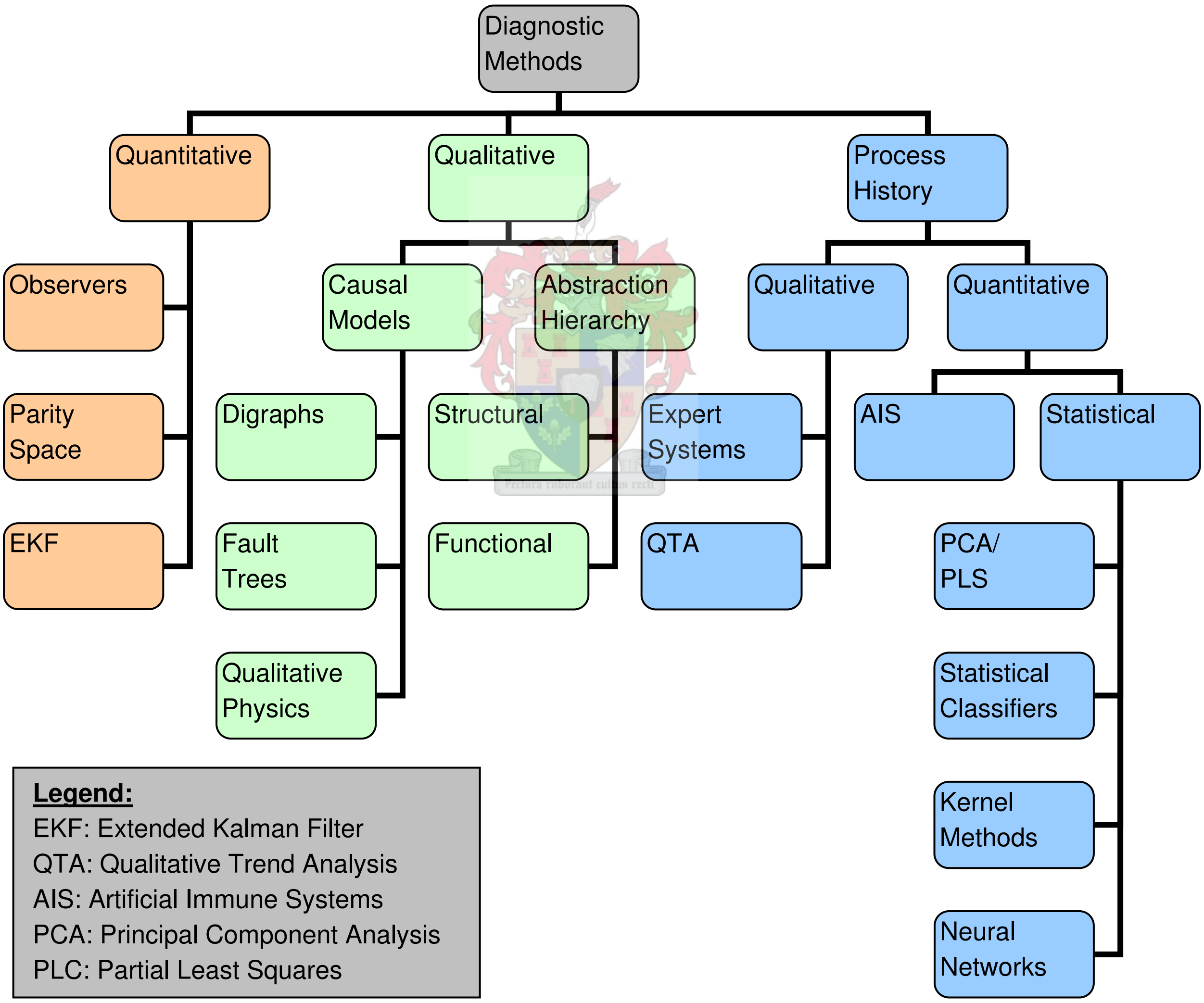
Despite recent advances in computerised control of process plants, troubling events such as the explosion at the Kuwait Petrochemical's Mina Al-Ahmedi refinery in June 2000 still occur. Two of the most devastating chemical plant accidents, namely Union Carbide's Bhopal, India and Occidental Petroleum's Piper Alpha, also occurred in recent years (Lees, 1996). Further, industrial statistics have shown that even though major accidents occur infrequently minor accidents can occur on a daily basis. This results in numerous occupational injuries, costing society billions of dollars annually (Bureau of Labour Statistics, 1998; McGraw-Hill Economics, 1985; National Safety Council, 1999). It can therefore be said that some room for improvement still exists which will not only save money, but also eradicate that feeling of personal commiserations after an accident.

1.1 Current Approaches to Fault Detection and Identification

The term *fault* is generally defined as a departure from an acceptable range of an observed variable or a calculated parameter associated with a process (Himmelblau, 1978). Detecting these faults forms a field of its own, i.e. Process Fault Detection and Identification. The field of process fault detection and identification can be sub-divided into three general categories: quantitative model based methods, qualitative model based methods and process history based methods (Venkatasubramanian et al., 2002a,b); refer to Figure 1. Model-based approaches can be broadly classified as qualitative or quantitative approaches. The model is usually developed based on some fundamental understanding of the physics of the process. In quantitative models this understanding is expressed in terms of mathematical functional relationships between the inputs and outputs of the system. In contrast, in qualitative model equations these relationships are expressed in terms of qualitative functions centred on different units in a process.

In contrast to the model-based approaches where prior knowledge of the model (either qualitative or quantitative) of the process is assumed, in process history based methods, only the availability of a large amount of historical process data is assumed. There are different ways in which this data can be transformed and presented as prior knowledge to a diagnostic system. This is known as the feature extraction process from the process history data and is done to facilitate later diagnosis. This extraction process can either proceed as quantitative or as qualitative feature extraction.

Although formal discussions on all of these methods are beyond the scope of this thesis, a basic comparison of some of these methods is given in Table 1. A brief discussion regarding some of the methods shown in Figure 1 will also follow.



Legend:
EKF: Extended Kalman Filter
QTA: Qualitative Trend Analysis
AIS: Artificial Immune Systems
PCA: Principal Component Analysis
PLC: Partial Least Squares

Figure 1: Classification of Diagnostic Algorithms

From Table 1, it is clear that there is (in theory at least) still some room for improvement on diagnostic methods. Any one of the methods shown in Figure 1 meets no more that 60% of the mentioned criteria and *no method can give a prior indication of the classification error made*. This sets a need for an improved system. In developing an artificial immune system, it is hoped to provide potential resolution to some of the limitations of current systems.

1.2 Desirable Features of Fault Diagnostic Systems

The comparative criteria used in Table 1 are now explained in order to clarify their meaning (Venkatasubramanian et al., 2002a):

- Fast Detection and Diagnosis

One would prefer the diagnostic system to respond quickly in detecting process anomalies. However, quick response and robustness are two conflicting goals (Willsky, 1976). A system that is designed to respond quickly is sensitive to high frequency influences and is therefore susceptible to noise.

Table 1: Comparison of various diagnostic methods (Venkatasubramanian et al., 2002)

	Observer	Digraphs	Abstraction Hierarchy	Expert Systems	QTA	PCA	Neural Nets
Fast detection	Yes	No	No	Yes	Yes	Yes	Yes
Isolability	Yes	No	No	Yes	Yes	Yes	Yes
Robustness	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Novelty Identifiability	No	Yes	Yes	No	No	Yes	Yes
Classification Error Est.	No	No	No	No	No	No	No
Adaptability	No	Yes	Yes	No	No	No	No
Explanation Facility	No	Yes	Yes	Yes	Yes	No	No
Modelling Requirement	N/A	Low	Low	Low	Low	Low	Low
Storage and Computation	OK	N/A	N/A	OK	OK	OK	OK
Multiple fault Identifiability	Yes	Yes	Yes	No	No	No	No

- Isolability

The isolability of a diagnostic system is its ability to classify different types of failures. This helps in the effective control and corrective action taken to control processes.

- Robustness

A diagnostic system should be robust to various noise and uncertainties. In the presence of noise, thresholds may have to be chosen conservatively, which slows down system response to anomalies.

- Novelty Identifiability

One of the basic requirements of a fault diagnostic system is the ability to classify between normal and abnormal process behaviour. If the system can distinguish between a known and unknown (novel) malfunction, that system is said to possess the ability of novelty identifiability.

- Classification Error Estimate

The reliability of a diagnostic system is measured by the error made during classification between normal and abnormal process behaviour. If the diagnostic system can provide an estimate of the possibility of a false classification occurring, the confidence intervals for decisions can be predicted. This could increase the user's confidence in the system substantially.

- Adaptability

In general, processes change due to changing external inputs or structural changes due to equipment wear, etc. A meticulous diagnostic system would allow for these changes and will continue functioning regardless thereof.

- Explanation Facility

When designing on-line decision support systems, one would like the diagnostic system to provide information on how the process fault occurred and propagated to the current situation. This requires the ability to reason about causality in the process.

- Modelling Requirements

The amount of modelling required for the development of a diagnostic classifier is an important issue. One would like to minimise the modelling effort for fast and easy development of real time diagnostic classifiers.

- Storage and Computational Requirements

Real-time systems would usually require algorithms that are computationally less complex, but might entail high storage requirements. The ideal diagnostic system would find a balance between these two competing requirements.

- Multiple fault Identifiability

A crucial yet difficult requirement of any diagnostic system is to be able to identify multiple faults. The interactions of multiple faults may prohibit the effectivity of a diagnostic system.

1.3 The decision-making process

After mentioning these criteria, it is helpful to view the diagnostic decision making process as a series of transformations or mappings of process measurements. Figure 2 shows the various transformations that process data undergo during diagnosis.

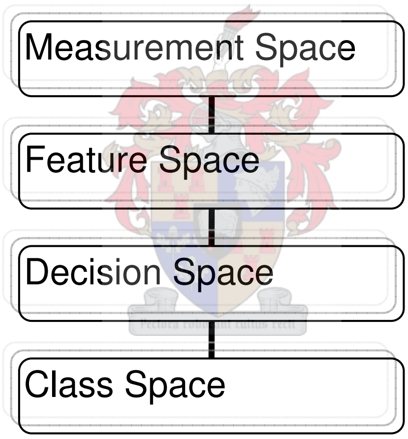


Figure 2: Transformations in a diagnostic system. (Venkatasubramanian et al., 2002a)

The measurement space is a space of measurements, $Y = (y_1,\ldots,y_n)$ with no historical process knowledge relating these measurements. They are simply the real values gathered from process sensors and form the input to the diagnostic system.

The feature space is a space of points, or events, $S = (s_1,\ldots,s_i)$, where s_i is the i^{th} feature obtained as a function of the measurements by using historical process knowledge. The transformation between measurement space and feature space is done because features generally cluster better in the feature space than measurements do in the measurement space (Venkatasubramanian et al., 2002a). This facilitates improved classification. Both of these spaces therefore contain data, with the basic difference that data is mapped or embedded into features in the feature space. There are two ways of coding the feature space from the measurement space, namely feature selection and feature extraction. In feature selection, a few important measurements are simply selected from the measurement space. Feature extraction, on the other hand, evolves a procedure that facilitates a transformation of the measurement space. An example would be to embed the measurement space, using process knowledge to determine the embedding-lag and dimensionality.

The decision space is a space of points D = (d_1,..,d_k), where k is the number of decision variables. They are referred to as detectors in the case of an artificial immune system. The transition between the feature space and decision space is achieved by either using a discriminant function, or a threshold function. This transition is implemented as a search or learning algorithm. For a strong or well-written learning algorithm, the prior knowledge need not be powerful. This means that the process under consideration needs not be well understood. A good learning algorithm will therefore provide the diagnostic system with valuable robustness.

The class space is a set of integers C = (c_1,…,c_m), where m is the number of failure classes, including normal. The transformation from decision space to class space is performed using threshold functions, template matching or symbolic reasoning. The class space is the final interpretation of the diagnostic system delivered to the user. It provides an index, classifying each event or feature into a class representing either an anomalous or normal event.

As an example, consider the Bayes classifier for a two-class problem. When assuming Gaussian density functions for the two classes, the Bayes classifier is developed as follows (Fukunaga, 1972): Measurements *x* are first transformed using a priori model information into features *y*. These features are then transformed into the decision space - which is a set of real numbers indexed by fault classes. The real number corresponding to fault-class *i* is the distance (d_i) of feature y from the mean (m_i) of class *i*. This distance is scaled by the covariance (Σ_i) of class *i*. For a two class problem, we have:

$$d_1 = (y - m_1)^T \sum\nolimits_1^{-1} (y - m_1) \qquad [1a]$$

$$d_2 = (y - m_2)^T \sum\nolimits_2^{-1} (y - m_2) \qquad [1b]$$

Where [d_1 , d_2] spans the decision space as *x* spans the measurement space. A discriminant function *h* maps the decision space to class space. Consider feature *y(i)* relating to measurement *x(j):*

h(i) = d_1(i) – d_2(i)

if h(i) < δ then x(j) belongs to class I

if h(i) > δ then x(j) belongs to class II δ is the threshold of the classifier and is the log of the ratio of the covariance of class II over class I, i.e.:

$$\sigma = \log\left(\frac{\sum_2}{\sum_1}\right) \qquad [2]$$

The Bayes Classifier is only one type of diagnostic system. As seen in Figure 1, there are many different types of fault diagnostic and identification systems. Examples of some of these methods will provide a better understanding to both the working and the need for an advanced new method in process fault detection and identification. Four methods were chosen from those shown in Figure 1 for reasons depending on their application. The first two methods under consideration are qualitative trend analysis and quantitative feature extraction. The rationale behind choosing these two methods is that they are both capable of analysing *dynamic* processes, which is the focus of this thesis. These two methods form examples of both qualitative and quantitative methods. Principal component analysis and neural networks are statistical methods widely applied in the field of diagnostic monitoring. They serve as good examples of the standard of today's systems that deal with historic process data. Principal component analysis is by far the most important technique and it is therefore sensible to provide background on this method.

1.3.1 Qualitative Trend Analysis (QTA)

Trend analysis and prediction are important aspects of process monitoring and supervisory control. Trend modelling is, amongst others, used to explain various important events happening in the process, to do malfunction-diagnosis and to predict future states. From a procedural perspective, a filtering mechanism (such as an autoregressive filter) is employed to prevent signal noise disrupting the signal trend. However, filters usually fail to distinguish between a transient and true instability (Gertler, 1989).

Qualitative trend representation often provides valuable information that facilitates reasoning about the process behaviour. In a majority of cases, process malfunctions leave a distinct trend in the sensors monitored. These distinct trends can be suitably utilised in identifying the underlying abnormality of the process. Thus, a suitable system can classify the process trends and detect the fault early on, thus leading to fast control. Such a system is trend triangulation (Cheung and Stephanopoulos, 1990). In this method, each segment of a trend is represented by its initial slope, its final slope and a line connecting the two critical points. A series of triangles then constitute the process trend. The actual trend now lies within bounding triangles, which illustrates the maximum error in the representation of the trend.

At a lower level, a pattern classification approach like neural networks is used to identify the fundamental features of the trends. At a higher level however, syntactic information is abstracted and represented in a hierarchical fashion with an error correcting code smoothing out the errors made at the lower level. Multilevel abstraction of important events in a process is possible through scale-space filtering (Marr and Hildreth, 1980). A bank of filters is used, each sensitive to certain localised regions in the time-frequency domain. There are two concepts in the use of multilevel abstraction of process trends. Firstly, changes in trends occur at different scales. Their optimal detection therefore requires the use of filters or operators of different sizes. Secondly, a sudden change will result in a peak or trough in the first derivative and a zero crossing in the second derivative. Zero-crossings are exceptionally rich in information regarding the changes in a trend. This gives rise to a necessity for filters whose characteristics are twofold: it should firstly be a differential operator and secondly, its scale should be adjustable. The Gaussian filter is an example of such a filter.

1.3.2 Quantitative Feature Extraction

Quantitative feature extraction generally regards fault detection and identification as a pattern recognition problem. When a process is under control, observations have probability distributions corresponding to the normal mode of operation. These distributions change when the process is disturbed or becomes out of control. Probability distributions are primarily characterised by their parameters if a parametric approach is used. Consider a normal distribution for a monitored variable; the parameters of interest are then its mean and standard deviation. Under faulty conditions, either the mean or standard deviation (or both) will deviate from the norm. If these deviations can be detected timeously, fault diagnosis can be achieved in both static and dynamic systems.

In on-line detection, decisions are made based on sequential observations. When an observation x(t) = [x_1, x_2,…,x_n] ε R_n, where R_n is the so-called stopping region in statistics, it can be said that a change has occurred in the process. A more common approach is to consider a function of the observations, g(t) and compare it with a given threshold function, c. When g(t) becomes larger than c, a change has occurred. It is clear that a well defined function, g(t) and a properly chosen threshold, c, may provide quick and accurate fault detection and identification. If these factors are poorly designed, fault detection and identification will almost certainly be irregular and ineffectual. There is, however, a trade-off in the design of these quantities, due to the occurrence of process noise. As the sensitivity to real change in the process is increased, for instance by altering the threshold value, the sensitivity to noise also increases. Hence, as the delay of the diagnostic system is tuned down, the number of false alarms will increase. A proper fault detection and identification system is one that will minimise both the false alarms and the delay according to the process employed. Some processes might require a precise classification, i.e. a low false alarm rate. Others might value quick response higher than faultless classification.

Although the basic principles behind Statistical Process Control (SPC) charts are still valid, the methods used to implement them cannot accommodate the progress in data acquisition technology. Additionally, the univariate control charts cannot handle correlation and may therefore be misleading. Multivariate techniques offer the capability of compressing data and reducing its dimensionality. These methods are described below.

1.3.3 Principal Component Analysis (PCA)

The PCA method was initially proposed by Pearson (1901), but was developed by Hotteling (1947) and has since been published in many textbooks (Anderson, 1984; Jackson 1991) as well as research papers (Wold, 1978; Wold et al., 1987). It is a multivariate technique, based on an orthogonal decomposition of the covariance matrix of the process variables along directions that explain the maximum variation of the data. The main purpose of PCA is dimensional reduction by maintaining the major trends of the original dataset.

Consider a process consisting of two variables, shown in Figure 3 (a). It is apparent that the data, enclosed by an ellipse, vary more on the one axis of the ellipse than the other. When performing a principal component transformation on the data, the first principal component is the direction of greatest variance in the dataset. The x-y axes are in a sense “turned” to coincide with the longitudinal axis of the ellipse. This results in the most important information in the dataset being explained by one dimension, rather than two as is the original case. This concept can be applied to larger dimensionalities, i.e. reducing a 5-dimensional system into say 3 dimensions.

Let X be a (n x p) matrix representing the mean-centred and autoscaled measurements with a covariance matrix Σ; where n is the number of samples and p is the number of measured process variables. From matrix algebra, Σ may be reduced to a diagonal matrix, L by an orthonormal (p x p) matrix, U, i.e. Σ = ULU’. The columns of U are called the principal component loading vectors, while the diagonal elements of L are ordered eigenvalues of Σ. They define the variance covered by each corresponding eigenvector. The principal component transformation is done as follows:

$$T = XU \qquad [3]$$

X is decomposed by PCA as follows:

$$X = TU' = \sum_{i=1}^{p} \theta_i u'_i \qquad [4]$$

The principal component scores are contained in the matrix T = (θ_1, θ_2,…, θ_p) and are defined as the observed values of the principal components for all *n* observations. The vectors θ_i are uncorrelated, seeing as the covariance of T is a diagonal matrix. Furthermore, θ_i and u_i are arranged in descending order according to their associated eigenvalues. It is found that the major trends in the data are usually described fairly accurately by the first few (two or three) principal components. This reduces the principal component decomposition to:

$$X = \sum_{i=1}^{a} \theta_i u'_i + E \qquad [5]$$

Where a is the number of principal components, with a < p and E is the residual term. This reduces the dimensionality of the system and allows one to consider only the uncorrelated variables in that system. Seemingly complex situations are simplified without losing any significant information.

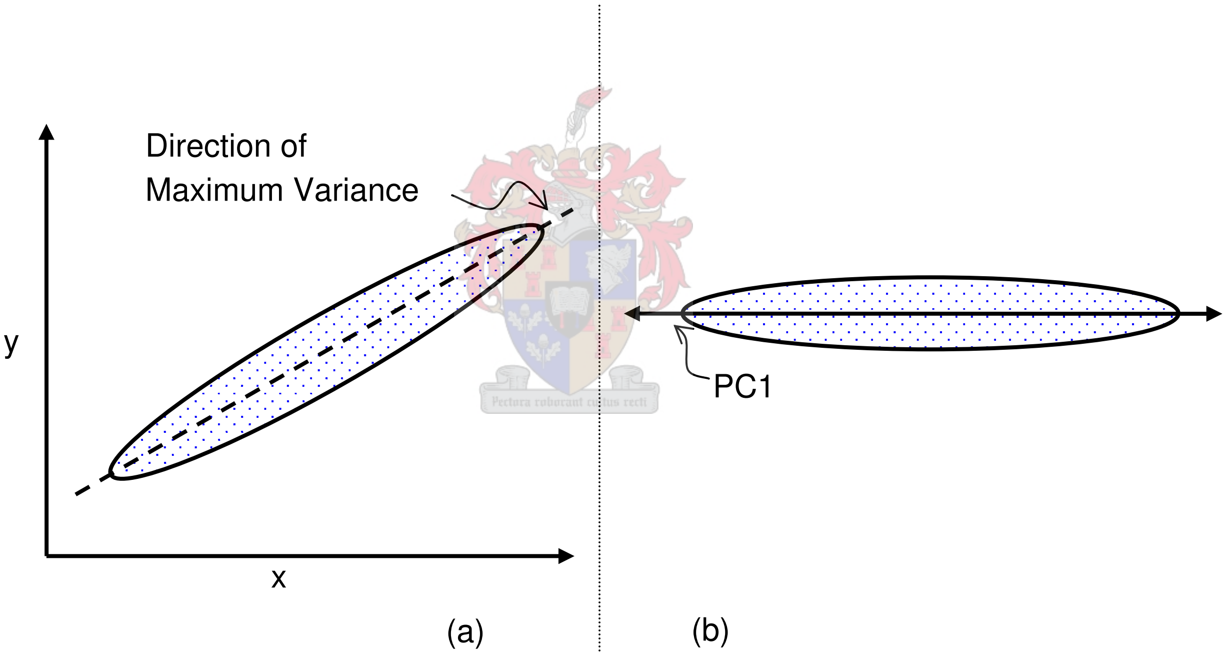


Figure 3: Principal Component decomposition of a 2-Dimensional ellipsoidal dataset. (a) is a two-dimensional dataset represented in its two dimensions: x and y; (b) is that same dataset represented in one principal component dimension: PC1.

1.3.4 Neural Networks

Neural networks are often used to extend PCA and extensive research has been done in this field (Venkatasubramanian, 2002b). Different network architectures have been used for the problem of fault diagnosis. One difference in these architectures is the method used for learning. One has the option between supervised and unsupervised learning. Supervised learning has taken the role as most popular strategy (Cybenko, 1998). The back-propagation algorithm is an example of this technique, whereas the ART2 network (Carpenter and Grossberg, 1998) is an example of unsupervised learning. The latter are also referred to as self-organising networks and the structure is adaptively determined due to the inputs to the network. Supervised learning uses a technique that determines the connection-weights explicitly by considering the difference between the desired and actual values. A detailed discussion of neural networks will not be given here, as abundant literature is available on this topic. The basic working thereof is however given.

An Artificial Neural Network basically consists of three elements: a layer of input nodes, a layer(s) of hidden nodes and a layer of output nodes. Every node in a layer is connected to every node in the next layer as shown in Figure 4(a). These connections carry with them a certain weight. This weight is the property that is varied during training.

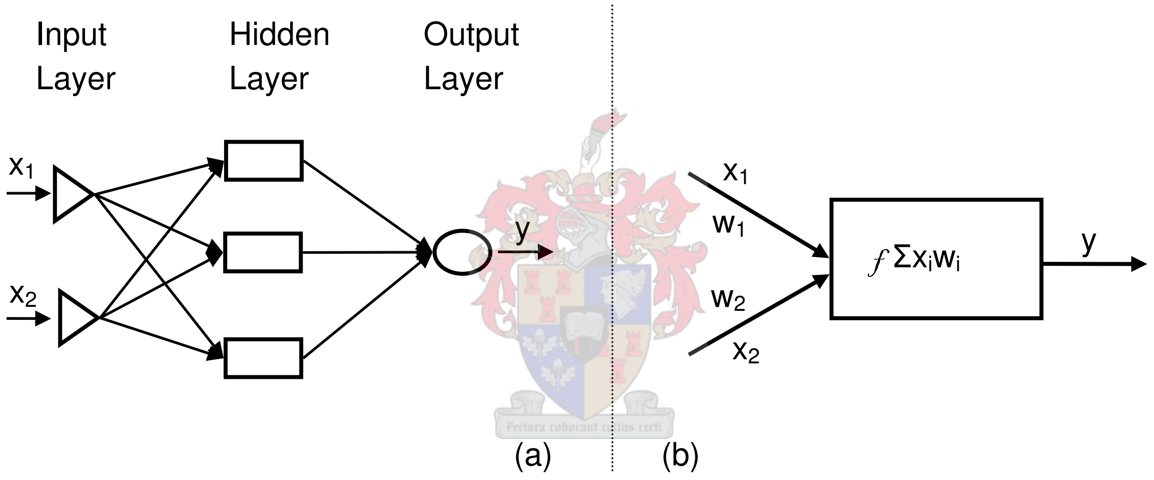


Figure 4: (a) Representation of a feed forward Neural Network; (b) Activation function inside the nodes

The number of nodes in the hidden layer and the number of hidden layers themselves are properties that are predetermined for every network. These are chosen by the user and are kept constant during training of the network. The activation function within the nodes is also predetermined. These parameters constitute the network's architecture.

A neural network learns by comparing the predicted output with an expected output or validation set, as shown in Figure 5. During training, weights are continuously updated in order to minimise the error, every iteration is called an epoch. Some networks will converge faster than others, depending on the transfer function and number of hidden nodes and hidden layers.

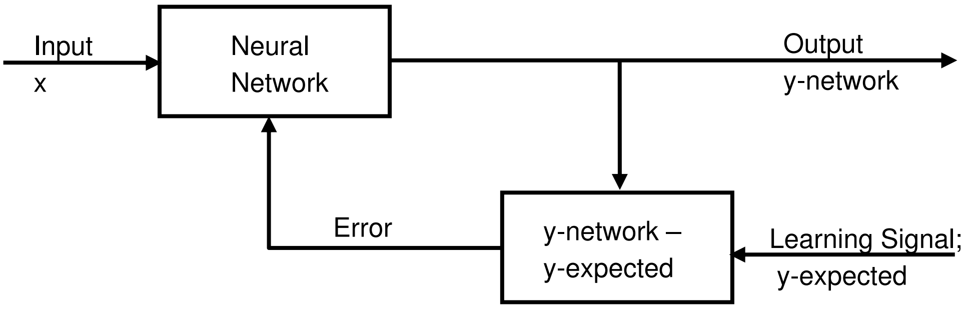


Figure 5: Supervised network, with backpropagation learning rule.

In unsupervised learning, the network is given inputs, but desired outputs are provided. The network must then decide which features it will use to group the input data. This is referred to as self-organising and may involve competition or co-operation between neurons. The training task is to group together patterns in the signal that are in some way similar and to then extract features of the independent variables.

For a collection of papers on the application of neural networks in chemical engineering problems, refer to Venkatasubramanian and McAvoy (1992).

1.4 Objectives of This Project

The main objective of this thesis is to asses the hypothesis that AIS can be a viable alternative to existing approaches for fault diagnosis in *dynamic process systems*. This objective will be met by considering the following sub-objectives:

- To conduct a literature review on the application of artificial immune systems in process monitoring.
- To develop an algorithm for automated fault diagnosis in dynamic process systems based on negative selection.
- To validate the chosen methodology via simulated case studies.

1.5 Summary of Chapter 1

Fault diagnosis and identification is vital in all modern process facilities. A short overview on a selection of current approaches is given in this chapter, mentioning that none of them fully satisfy the favourable detection criteria. AIS are introduced to provide a potential solution to some of the problems arising in other methods.

This chapter discusses some of the current uses to fault detection and identification. It also looks at a Qualitative, Quantitative and a Statistical method. Further, a learning method, namely the Neural Network is considered, while attention is laid on certain design criteria that is favourable in a diagnostic system.

Chapter 2: Artificial Immune Systems

This chapter considers both the natural and artificial immune systems as well as the relationship between them. It is important to realise that Artificial Immune Systems do not strive to exactly mimic the natural immune system. Rather, it applies equivalent principals to existing computational abilities, in order to create a digitised system which imitates the functionality of the natural immune system. The differences between the natural and artificial immune systems become clear upon development of the presented text.

2.1 The Natural Immune System

In the natural immune system, the primary producers of both innate- and adaptive-immune responses are leukocytes (Luh and Cheng, 2005). From the several different types of leukocytes, there are two types that are of concern to this project: phagocytes and lymphocytes. Although not much is known about the exact working of the natural immune system, certain elements are clear (Eli et al., 2000). Phagocytes are the first line of defence for the innate immune system. These cells bind to a variety of organisms and destroy them. Phagocytes are positioned, by the immune system, at specific locations, where they are more likely to encounter organisms that they are most suitable to control (King et al., 2001). This is an astute method of managing limited resources. Lymphocytes, on the other hand, initiate adaptive immune responses and specifically recognise individual pathogens (King et al., 2001). The main categories of lymphocytes are B-lymphocytes (B-cells) and T-lymphocytes (T-cells) (Luh and Cheng, 2005). B-cells are produced in the bone marrow, while T-cells develop in the thymus (Yang et al., 2006; refer to appendix).

Figure 6 shows the receptive antibodies on the surface of B-cell. These are used to identify specific antigens.

Since the innate immune system is of lesser importance to this project, focus is placed on the adaptive immune system. Lymphocytes respond to a given pathogen by first identifying it. This is done by means of different mechanisms: the compliment system, cytokines and antibodies (Tarakanov and Dasgupta, 2001).

Cytokines are released by T-helper cells (a subclass of T-cells) and communicate with other cells in the immune system. In the innate system, phagocytes present pathogens to the T-helper cells to aid in antigen recognition. If the T-helper cell recognises the pathogen as an antigen, it will release cytokines to activate the phagocyte, which will in turn destroy the pathogen.

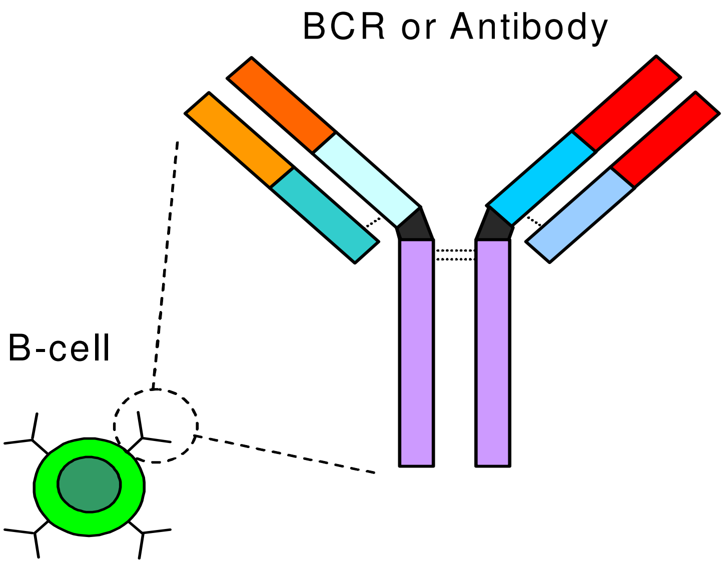


Figure 6: B-Cell receptor (BCR) or antibody, located on the surface of a B-cell.

B-cells manufacture antibodies. However, each B-cell is monoclonal (i.e. it can only manufacture one type of antibody) (King et al., 2001), see Figure 6 and is therefore limited to recognizing antigens that are structurally similar to itself. The paratope is the genetically encoded surface receptor on an antibody molecule (King et al., 2001). This is the region where recognition of the antigen takes place. The area on the antigen where a paratope can attach itself is known as the epitope (King et al., 2001), as shown in Figure 7. This gives rise to the so-called shape space affinity. When a particular antigen is recognized or detected by a B-cell, it combats the antigen by reproducing in numbers (i.e. cloning) and releasing its soluble antibodies to bind to the target antigen. It should be noted that an antibody produced by a B-cell is specific for an epitope and not the entire antigen molecule. In fact, several different antibodies from different B-cells may bind to a single antigen. The binding of an antibody to an antigen is a result of multiple non-covalent bonds (King et al., 2001).

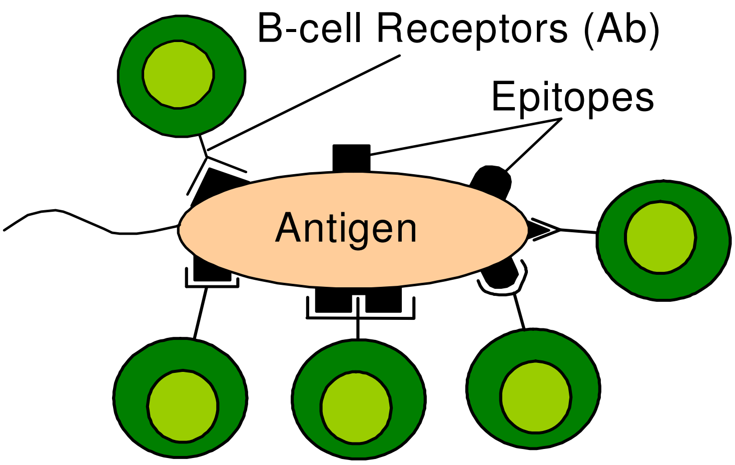


Figure 7: Antibodies of different B-Cells attach to different epitopes on an antigen

Antibody affinity is a measure of the strength in the bond between the antibody's paratope and a single epitope (King et al., 2001). Structurally similar epitopes will also bind to this antibody, but with lower energies, giving rise to the concept of an affinity threshold. Recognition has occurred when this threshold is exceeded. The B-cell is subsequently reproduced. It is however, not exactly reproduced. The "clones" undergo a slight mutation, ensuring that the immune system does not stagnate, but evolve to improve the immune response (Stibor et al., 2005). This is said, because the parent cell probably had a less-than-maximum affinity when it was stimulated. An exact clone could not improve the affinity, while a slightly mutated offspring at least has a chance of improvement. Only better-suited offspring will be stimulated to produce offspring of their own. Continuing this cycle ensures that subsequent generations recognise antigens effectively. The immune system therefore exhibits functionality of evolutionary adaptation.

This evolutionary system is called the immune system's primary response. A finite amount of time is required from the initial encounter of the antigen until the immune system has adapted its B-cells to best combat the intruder (Stibor et al., 2005). The immune system has a way of remembering a certain intruder in order to act faster on a future invasion. Antibodies are already available and need not be evolved from a base antibody. This is termed the secondary response of the immune system and is extremely rapid and specific (King et al., 2001).

One of the primary functions of the natural immune system is to perform classification of organisms as either self or non-self (i.e. your own versus foreign). This capability is distributed among a variety of agents (adaptive and non-adaptive) whose function it is to continually monitor the body's environment for these undesirable intruders. If an organism is classified as self, then no actions are taken. However, if an entity is classified as non-self, the immune system's agents work autonomously and/or in concert to eliminate the organism. In an artificial immune system, the object is not to eliminate a non-self event, but to accurately identify it.

2.2 The Artificial Immune System

Artificial immune systems employ the principles discussed above to monitor dynamic process systems for anomalies. Some of their applications are discussed in (Dasgupta, 1999), (Dasgupta and Forrest, 1995) and (Dasgupta and Forrest, 1999); a summary is given in section 2.2.4b. It is important to realise that an artificial immune system does not strive to perfectly match the working of the natural immune system; it is simply analogous towards it. As in the natural immune system, there are several key-players that coordinate the working of the artificial immune system. They each have a specific counterpart in the natural immune system that has, more or less, the same function.

Historical process data are used to create a unique space inside the shape space, called "self". This relates to the natural immune system's sense of "self". The transformation from time series data, to the self-space (or measured space to feature space, as described above) is discussed in detail in later sections. At this point, it is sufficient if the reader can associate a sense of self with the expected or normal behaviour of a process facility.

Detectors are generated in the AIS and are anomalous to either lymphocytes, or phagocytes, depending on how they are generated. Phagocytes are placed at specific locations, where antigens are expected (King et al., 2001). This relates to the hypercube vertex method for generating detectors in the AIS, since they are scattered around the self-space. Lymphocytes (B and T cells) relate to pseudo-randomly generated detectors. The various methods for generating detectors are discussed in Section 2.2.4 as well as Section 3.2. When a method has been selected and detectors have been generated, a threshold, similar to that in the natural immune system is introduced. This threshold is defined by the matching rule, discussed in Section 2.2.3.

Upon the occurrence of process variables drift, process anomalies occur. This is analogous to antigens in the natural immune system. It forms part of the non-self space in the shape space and is recognised as non-self by a natural immune system.

An immune response occurs when the detectors recognise an event as part of the non-self space. There are various different basic types of artificial immune systems and the type of immune system used determines the implication of an immune response. For instance, under the negative selection algorithm an immune response means that a process anomaly has occurred, since one or more detectors have identified an antigen. Under the positive selection algorithm, on the other hand, an immune response will also identify a process anomaly, but in this case the lack of association between the given event and any detector triggers a response. This basically means that a highlighted event under the negative selection theorem relates to a process anomaly, while under the positive selection theorem it relates to normal process behaviour.

2.2.1 Overview of Artificial Immune Algorithms

There are several types of Artificial Immune Systems, each of them having their own characteristic features. The most important types of AIS discussed in this section are Negative Selection, Clonal Selection, Immune Network Models and Danger Models.

a Negative and Positive Selection

One accepted mechanism for immunological self-nonself classification is called negative selection. The principle of positive selection has also been considered (Esponda et al., 2004; Garret, 2005) and its accepted process is very similar to that of negative selection allowing these two methods to be discussed simultaneously. As mentioned earlier, T-cells are generated in the thymus and undergo a natural classification process. The surviving T-cells detect invading nonself entities by matching epitopes to paratopes.

Applying this algorithm to the artificial realm of fault diagnosis and identification is a matter of regarding small sections of time, or *windows* of time as events in a certain space (Dasgupta and Forrest, 1995). T-cell like detectors are then generated by one of various different types of methods and monitor the activity of a process. They identify “invading” events in a similar way that a T-cell would bind to an antigen. A similar threshold exists in the matching rule, also discussed later.

In positive selection, detectors are placed inside the self-space. They identify events that fall close enough to themselves. As soon as a certain event passes unnoticed by the detectors, it is termed anomalous to the system and an alarm is raised. The positive selection algorithm is usually effective for situations where the self-space is small and well defined. It is also useful in binary alphabets, e.g. network intrusion detection (Esponda et al., 2004). However, in dealing with process systems, a binary alphabet will only provide data with “high” and “low” readings. This is rarely adequate in industrial processes, as process variables are usually real values.

The negative selection algorithm places detectors around events in the self-space. It is capable of treating large datasets in real valued alphabets. It is therefore more suitable to the field of process fault diagnosis and identification than, for instance, the positive selection algorithm. For this reason, the negative selection algorithm is the chosen method in this project, therefore shaping the focus on subsequent discussions. A formal discussion of negative selection, of which the detail is in excess of the scope of this section, is given in section 2.2.2.

b Clonal Selection and Hypermutation

The natural immune system’s ability to adapt its B-cells to new types of antigen is done by means of two processes, called *clonal selection* and *affinity maturation by hypermutation*. Garret (2005) states that biological clonal selection is dependent on the degree that a B-cell matches an antigen. A strong match will result in the B-cell being cloned rapidly, with small mutations occurring, while a weaker match will cause the B-cells to clone slower - undertaking larger mutations. This results in the formation of large amounts of strong B-cells, where none of them mutate too far from the given antigen. Weaker B-cells undergo larger mutations, striving towards a better match. They are present in smaller numbers in order not to flood the immune system with weak B-cells.

The artificial form of clonal selection has been popularized mainly by de Castro and von Zuben (2002). Their algorithm, CLONALG, performs two tasks: Optimisation and pattern matching. The optimisation algorithm covers multiple optima at the same time (multi-modal optimisation). This is done by mutating a given set of detectors and constantly adding random detectors. Mutation is controlled by a certain fitness that is assigned to the detectors (de Castro and von Zuben, 2002). The pattern-matching algorithm works similar to the optimisation algorithm, with one difference: when training the system, the member with the highest fitness of each population is compared to the member with the overall highest fitness thus far (champion). If the current population's champion has a higher fitness than the overall champion thus far, the previous champion is replaced. The method basically follows a Genetic Algorithm approach. Garret (2005) reports that the main drawback of these methods is scalability. Since the number of clones rises in proportion to the value of f(ab), the number of objective function evaluations will quickly increase as the majority of the population approaches the (local and global) optima. This causes the algorithm to run very slowly.

c Artificial Immune Networks

As mentioned earlier, an antibody will bind with an antigen by means of receptors on their surfaces, named paratopes (on antibodies) and epitopes (on antigens). Immune network models state that antibodies also have epitopes which binds to other antibodies' paratopes. The entity presenting the epitope, be it an antibody or antigen, is then eliminated, while the antibody presenting the paratope is reproduced (Garret, 2005). A network of stimulatory and suppressive interactions exists between antibodies. This interaction effects the concentrations of each type of antibody and according to Garret (2005); it has been shown that this might allow for associative memory.

Artificial Immune Networks are described by two equations, one defining the matching affinities (m_{ij}) between antibodies and the other describing the change in concentration (x_i) of antigen types (Farmer et al., 1986). Both antibodies and antigen are first modelled as binary strings. Once this is done, the matching affinities are given by Equation 6 (Garret, 2005):

$$m_{ij} = \sum_{k=1}^{rng} G\left(\sum_{n=1}^{l} e_i(n+k) \oplus p_j(n) - s + 1 \right) \qquad [6]$$

The "$\oplus$" operator is the complimentary XOR (exclusive or) operator and explains the use of a binary alphabet for representing antibodies and antigen. In Equation 6, k is the offset measured in bits between the paratope and epitope; $e_i(n)$ is the n_{th} bit of the epitope; $p_j(n)$ is the n_{th} bit of the paratope; s is a threshold since G(x) = x if x > 0 and G(x) = 0 otherwise. The number of bits in a string, $l = \min(length(e_i), length(p_j))$; while $rng = l - s; \qquad s \leq l$.

Given N antibody types, with concentrations $\{x_1,..x_N\}$ and n antigen types, with concentrations $\{y_1,..y_n\}$, the change in concentration of a certain x_i is given by (Garret, 2005):

$$\frac{dx_i}{dt} = c\left[\sum_{j=1}^{N} m_{ji} x_i x_j - k_1 \sum_{j=1}^{N} m_{ij} x_i x_j + \sum_{j=1}^{n} m_{ji} x_i y_j\right] - k_2 x_i \qquad [7]$$

The $x_i x_j$ and $x_i y_j$ elements model the probability that a paratope and epitope will be close enough to attempt to bind, since high concentrations of either will increase this probability. The first term in Equation 7 models the stimulation of an antibody type's paratope binding to the epitope of another type of antibody, as indicated by the double x's. The second term also contains only x's, but this term models the repression (hence the negative sign) of an antibody type due to its epitope binding with the paratope of another antibody, thus the inverted m_{ij} as apposed to m_{ji} in the first term. The constant k_1 ($k_1 > 0$) indicates a possible inequality between the stimulation and repression terms. The third term describes the stimulation of an antibody type's paratope binding with the epitope of an antigen. A so called "death term" is added to the equation; $k_2 x_i$ ($k_2 > 0$), this removes a quantity of x_i antibodies. Finally, c is a rate constant that depends on the number of collisions per unit time and the rate of antibody production stimulated thereby.

This model has one major negative implication: It is restricted to the use of binary data and debatably unsuitable for binary representations of real-valued data. This is said because the binary strings for 63 [1 1 1 1 1 1] and 31 [0 1 1 1 1 1] only differ in one bit, but in base_{10}, these values are clearly not related.

With all this being said, the method of choice for this project remains the Negative Selection Algorithm. Though it has been briefly mentioned earlier, it should become clear that this method is best suited for real valued, dynamic systems with larger datasets in more dimensions. Typically, industrial processes match this description rather well. A detailed description of the Negative Selection Algorithm thus follows.

d Danger Theory

The latest theory is the danger theory, which is based on recent modifications to the self-nonself theory. These modifications were made in an attempt to explain why there is no immune response to bacteria in our intestines or the air in our lungs, both of which are clearly nonself. According to Garret (2005), it was suggested that the immune system may require more than the detection of a nonself element to induce a response. A second requirement might be that of cells being under some sort of stress, such as a viral or bacterial attack. Thus, an immune response will only occur if [a] a nonself entity is identified *and* [b] an attack or cell death is noticed. This helps prevent autoimmunity.

The main advantage of Danger Theory Artificial Immune Systems (DT-AIS) is that it eliminates most of the false alarms generated during monitoring. It is however limited to systems of which one has a good understanding, not only of the nonself space, but also the dangerous nonself space. This means that initially the system needs to expose the process to so-called dangerous anomalies, in order to be able to identify them in the future. In applications such as computer antivirus software this is not a problem, since exposing the computer to signals such as high memory or disk activity or other unusual signals is not detrimental. In an industrial process however, consider exposing a nuclear reactor to a dangerous process anomaly. Causing a nuclear meltdown in order to prevent one in the future seems rather impractical.

2.2.2 The Negative Selection Algorithm

Forrest et al. (1994) first published the negative selection approach to anomaly detection. Since then, interest in negative selection has been rapidly growing. In the negative selection algorithm, detectors are generated in the shape space, around but not on normal events, or “self” events. This means that, given a certain shape space U, a self-set S and a nonself set N, then:

$$U = S \bigcup N \qquad and \qquad S \bigcap N = 0$$

The Negative Selection Algorithm is summarized in Figure 8. Detectors are generated in the shape space, around the self-events. Methods for generating these detectors are discussed in Section 3.2. The detectors are then assigned with a certain threshold. The threshold adjusts the sensitivity of the detectors to intrusion. As soon as any given event pervades this threshold’s perimeter, an anomaly is identified. The detectors are placed at a given distance from the self-space. This distance can be adjusted in order to allow for sensor noise etc. These two detector properties are managed by means of a matching rule. Esponda et al. (2004) stated that there are two key factors in creating a successful artificial immune system: the choice of matching rule and the method for generating detectors. There are various different methods of managing these tasks and will be discussed in the following subsections.

The notion of shape space was introduced by Perelson and Oster (Stibor et al., 2005) and allows a quantitative affinity description between antibodies and antigen. A shape space is in effect a metric space with an associated distance function or affinity function. The Hamming-space and real-valued shape-space are most commonly used in Negative selection and will be briefly discussed hereafter.

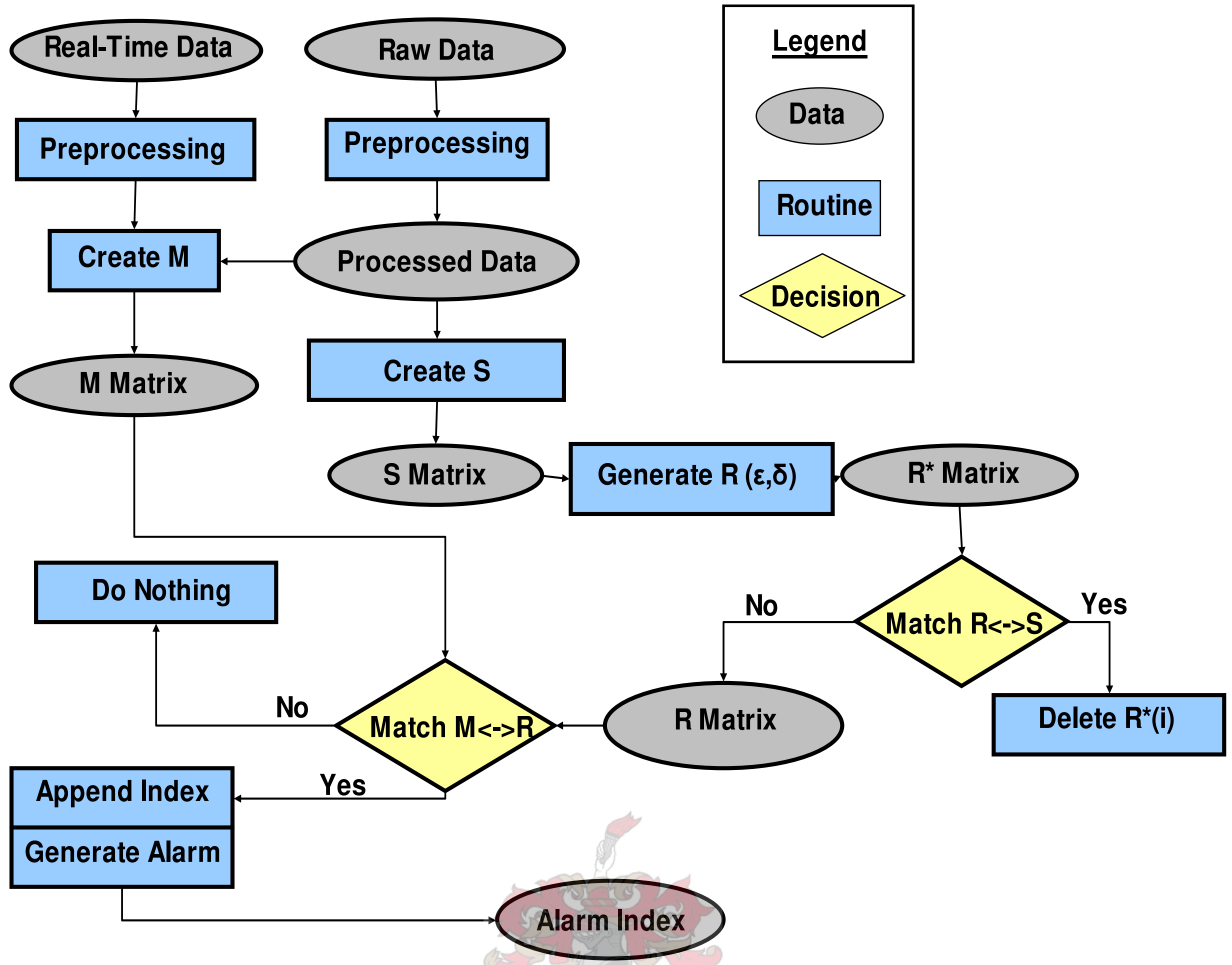


Figure 8: The Negative Selection Algorithm

a Hamming Shape-Space and r-Chunk matching

The Hamming shape-space U_l^{Σ} consists of all elements of length l defined in a finite alphabet Σ. Forrest et al. (1994) defined this space in a binary alphabet Σ {0,1}. The r-chunk matching rule achieved the highest matching over a binary alphabet (Stibor et al., 2005). The r-chunk rule is an improved version of the r-contiguous bits rule, which states that two strings will match if they contain at least r-contiguous matching bits. Formally, the r-chunk rule is defined as:

Given a shape space U_l^{Σ}, which contains all elements of l over a finite alphabet Σ and another shape space D_r^{Σ} representing a number of detectors in U_l^{Σ}. By definition of r-chunks, an element e ε U_l^{Σ} with e = [p,e$_1$, … ,e$_l$] and detector d ε D_r^{Σ} with d = [d$_1$, … ,d$_r$] for $r \leq l$ and $p \leq l - r + 1$, will match if e$_i$ = d$_i$ for i=p,..,p+r-1

Informally, element e and detector d matches if a position p exists, where all characters of e and d are identical over a sequence length r.

b Real Valued Shape-Space

Consider a real valued shape-space R_k^Δ that consists of elements of length k, over a real-valued alphabet Δ. Unlike the hamming shape-space, there is an infinite amount of elements that occupy this space. The self-space can therefore not be entirely covered by self-elements, as shown in Figure 9. Detectors fill a certain space D_k^Δ with elements d(c,r_d), where c is a set of coordinates with a dimensionality matching that of the shape space and a detector radius r_d ε $\mathbb{R}$ that can be adjusted to optimise detector efficiency. Figure 9 shows a two dimensional real-valued shape-space, with self-elements forming an attractor. Two sets of detectors are generated with different radii. If an element lies within a detector, i.e. the Euclidean distance d_E < r_d, that element is classified as nonself.

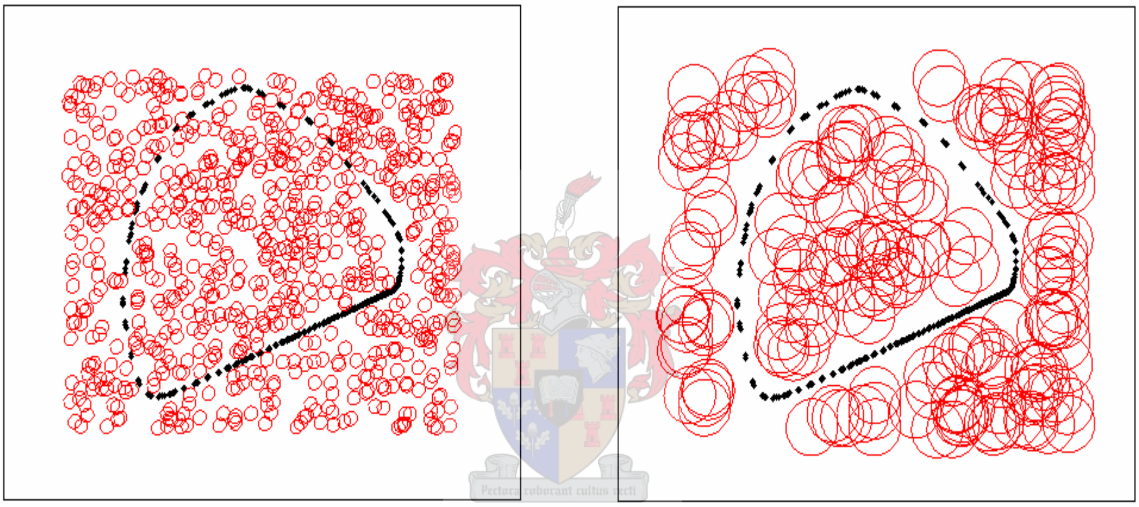


Figure 9: Real-valued shape-space, with self-elements (dots) and negative selection detectors (circles with radius r_d)

The radius r_d strongly depends on the probability density function of the data set (Stibor et al., 2005), which is unknown a-priori. An improper radius results in poor classification performance. This performance is measured by the ROC analysis, discussed in section 3.4. To estimate a proper radius by using only the self-data, a coherence between the estimated probability density function and detector radius must be found. Another problem is how to find optimal distribution of the detectors, i.e. the minimum number of detectors covering the maximum possible nonself space. As a consequence, a vast amount of time is needed to generate and to position random detectors to cover the nonself space. According to Stibor et al. (2005), the negative selection algorithm is inefficient since a vast number of randomly generated detectors are discarded before the required number of suitable detectors is reached. The fatal flaw in this statement is that random generation is not the only method to generate detectors. This project shows that the Negative Selection Algorithm is not only efficient, but also reliable and consistent when using appropriate methods to generate detectors.

2.2.3 Significance of Matching Rules

In statistical anomaly detection, the degree of suspicion associated with an event is indirectly proportional to the historical rate of recurrence of the specific event (Stibor et al., 2005). The main snag in applying statistical anomaly detection is that the probability distribution is unknown. A second family of approaches exist to anomaly detection. They attempt to define a measure of distance between events in the shape space. The Hamming Distance and r-chunks rules were already discussed and they form examples of these distance measures.

Under the distance approach, the degree of suspicion to an event is directly proportional to the distance from the event to some reference. This reference can be the nearest observed normal event, the nearest detector, or the centre of mass of the nearest cluster of normal events. The impediment in this approach is the definition of the distance measure. Choosing a measure that does not accurately describe the shape-space produces pointless classifications. This choice is made when choosing a matching rule. Studies have been done using the RCB matching rule and Hamming distance rules for partial matching (Dasgupta and Forrest, 1995; Esponda et al., 2004). Most of this work involved anomaly detection in computer systems; such as data transfer monitoring, or virus detection. These systems run in a binary alphabet, in which case rules like r-chunk and Hamming Distance describe the shape-space fairly accurately. However, applying these rules to dynamic process systems, which operate in a real alphabet, would result in insufficient representation of the shape space. A novel measure is therefore introduced in this project, namely the Euclidean Distance (ED) rule. It was found that partial matching rules somewhat obscure the boundary between the self and non-self sets. This places limits on the precision and coverage that is possible using most plausible matching rules.

A matching rule describes when two points in an n-dimensional space fall close enough together to trigger a response. It generates alarms when a process anomaly falls in the space of one or more detectors. It is also used during generation of the detectors themselves. When a detector is generated, it is tested against the known self-space to avoid autoimmunity. Two matching rules are considered at this stage, namely the r-contiguous bits rule (RCB) and the Euclidean distance (ED) rule, while r-chunks were discussed earlier. These rules all detect partial matching, as exact matching will require an infinite number of detectors to be generated.

a R-Contiguous Bits (RCB) rule

In the RCB rule, two strings match, if they are identical in *r* contiguous positions. Take two strings:

- a = 1 2 3 4 5 6
- b = 4 2 3 4 6 5

For *r* <= 3, these two strings match, because they have 3 adjacent matching positions. If *r* > 3, these strings will not match.

In the R-contiguous bits rule, the degree of detector sensitivity is determined by choosing *r*, keeping in mind that *r* is always smaller than or equal to the window size of the self-space and therefore also to that of the detectors.

b Euclidean Distance (ED) rule

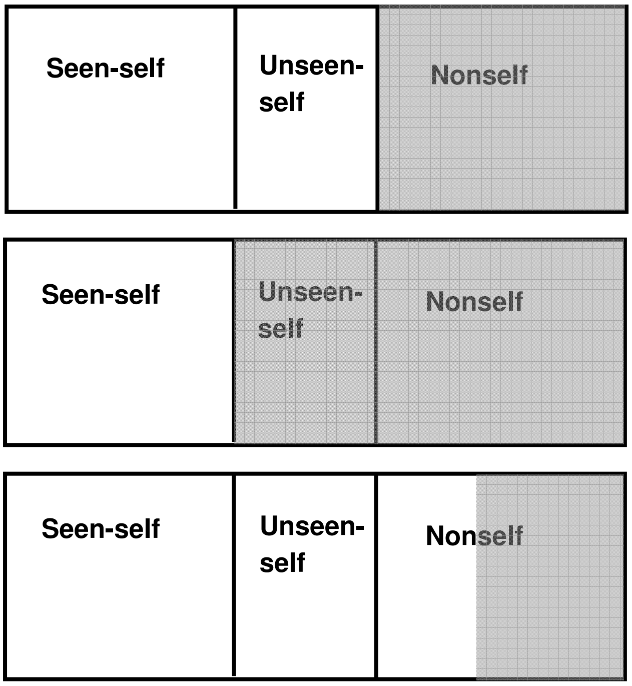
The Euclidean distance between two points (points *a* and *b*) in an *n*-Dimensional space is given by:

$$d_E = \sqrt{\sum_{i=1}^{n} \left(x_a - x_b\right)_i} \qquad [8]$$

Under the ED rule, two points will match if the Euclidean distance d_E is less than some threshold: $d_E < \varepsilon$.

c Generalisation by holes

All matching rules cause undetectable areas in the shape-space, due to their accommodating nature regarding imperfect matching. These areas are generally termed “holes” (Stibor et al., 2005) and are elements not seen during the training phase. No detectors exist or can be generated for these areas (under the given matching rule and affinity threshold) and can therefore not be identified as nonself. This is true because these holes exist in spaces between self-elements that are too small to accommodate a detector. Although it may sound that holes are a classifier imperfection, the contrary is true. Holes are essential in AIS and are one of the main factors that distinguish AIS from other classifiers. They allow the system to generalise beyond the training set. A detector set that generalises well ensures that both seen and unseen self-elements are not recognised by any detector, whereas all other elements are classified as nonself. This is illustrated in Figure 10.



[a] The detector set generalises well as all self-elements (seen and unseen) are classified as self.

[b] The detector set overfits, because no holes exist and therefore unseen self-elements are classified as non-self.

[c] The detector set underfits, because holes are too large and a number of non-self elements are misclassified as self.

Figure 10: Representation of classification by detectors: Grey areas are classified as nonself in the three examples above.

Esponda et al. (2004) approximates the number of holes given a self-set with size |S|, string length l and r-chunk detector length r by:

$$
\begin{aligned}
&H(|S|,l,r)=T_1\cdot T_2-|S| \\
&where: \\
&T_1=2^r-2^r\left(1-\frac{1}{2^r}\right)^{|S|} \\
&T_2=\left[2-0.5\left(1-\frac{1}{2^{r+1}}\right)^{4|S|}-2\left(\left(1-\frac{1}{2^{r+1}}\right)^{2|S|}-\left(1-\frac{1}{2^{r+1}}\right)^{3|S|}\right)\right]^{(1-r)}
\end{aligned}
\qquad [9\ a\text{-}c]
$$

When assuming that |S| < |N|, with S being self and N being nonself, the number of holes increases exponentially for r → l,…,1. The r-chunk method will therefore underfit exponentially. A linear under/overfitting will occur if r is close to l; however, this causes the detector generation to become non-feasible, since runtime complexity usually increases exponentially in r. Therefore, the Hamming shape-space and the r-chunk matching rule are only appropriate and applicable for anomaly detection problems for small values of l, e.g. 0<l<32 (Stibor et al., 2005).

2.2.4 Generating Detectors

Generally, detectors are generated offline and the entire self-set is known at the time of generation. Occasionally, a set of anomalous patterns is known or thought to be known entirely and a separate detector set is used to cover the “dangerous” parts of the nonself precisely. Examples of this approach are signature-based scanners for intrusion detection and commercial virus scanners.

Linear time series algorithms have been widely employed, using various threshold values. Wierzchon developed a low space-complexity algorithm for generating an optimal collection of detector strings (Wierzchon, 1999; Wierzchon, 2000; Wierzchon, 2002). Hofmeyer, on the other hand, used a random generation method for producing detectors (Hofmeyer and Forrest, 1999; Hofmeyer and Forrest, 2000). He reported difficulties in generating valid detectors and concluded that negative selection alone is not feasible for his specific application, which was network intrusion detection. Williamson (2002), however, reported positive results when using negative selection on the network intrusion detection problem.

The methods employed in this project are convex hulls, random generation and a new method called the hypercube vertex approach. These methods are described in full detail in the following chapter. However, a short summary of the random generation method as used in literature would be beneficial at this stage. Another method used in literature is the so termed hypermutation method. This method is also described below although it will not be used in this thesis mainly because it requires knowledge of the nonself space, which is accepted to be unknown in this project.

a Random generation

The simplest way of generating detectors, is to generate a random string, d of length l and values ranging within given bounds. These bounds are usually [0,1] for a systems in which the self-space S has been auto-scaled, but can be increased by any factor, e.g. 1.5 to accommodate for a larger nonself space.

There are two basic limitations that govern the detectors. These are:

- Detectors must not fall outside the range of the shape space, e.g. [0,1.5]
- Detectors must not fall closer than a given distance, ε to the closest event in the self space. Typical values of ε are in the range {0.05…0.20}.

A generated detector string d is then tested against all the strings in self-space S by means of the specific matching rule. If d, matches any string in S it is rejected, otherwise it is accepted into the group of detectors R. This process is repeated until R reaches a certain size, or a certain percentage of the nonself-space N is covered.

The number of detectors generated has certain implications. Generating too many detectors will result in high computational complexity and monitoring new inputs will take a long time. Too few detectors will result in a poor coverage of N, causing process anomalies to pass undetected.

Figure 11 shows an example of random detectors in a 3-dimensional shape-space.

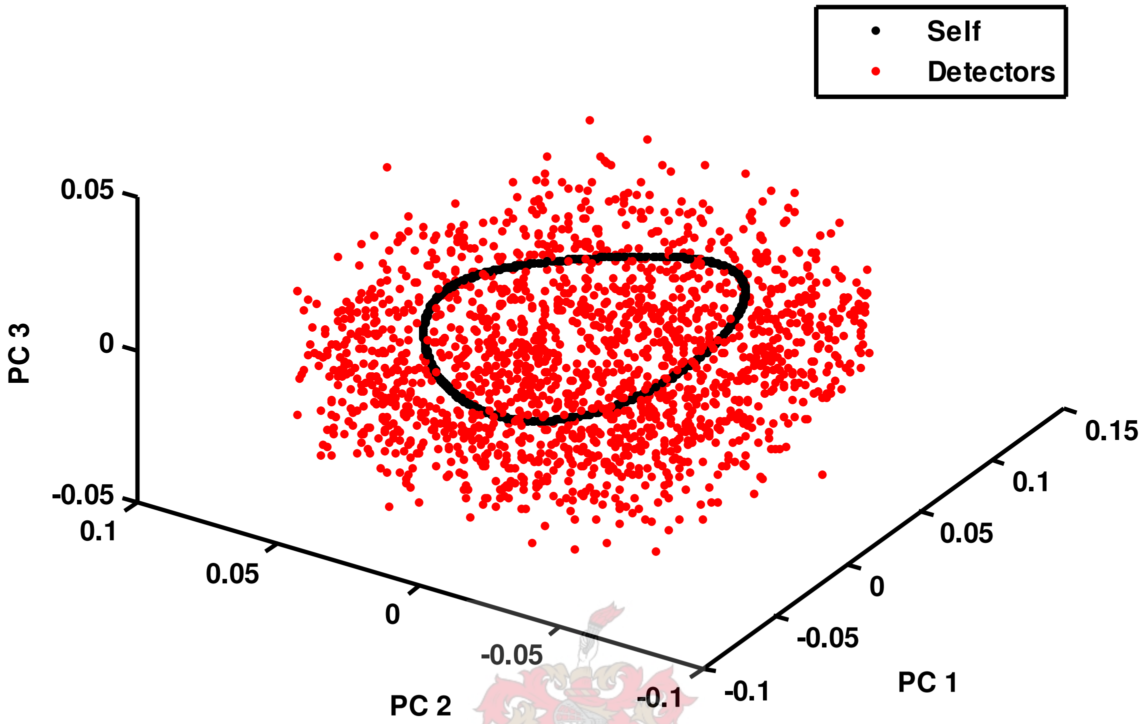


Figure 11: Randomly generated detectors in a 3-dimensional shape-space.

b Hypermutation

Analogous to the natural immune system where B-cells respond to particular antigens, each detector in the artificial immune system can respond to stimulus from a process anomaly (Timmis et al., 2000). In the natural immune system, the level to which a B-cell is stimulated relates both to how well its antibody binds to the antigen and to its affinity to its neighbours in the network (Farmer et al., 1986). AIS reflect all three of these influences.

The primary stimulus is the affinity between the detector and the pathogen (training data item), i.e. how well they match. This is defined by: $ps = 1 - pd$; where *ps* is the affinity between the two events; *pd* is defined as the distance between the two items in the normalized data space such that $0 < pd < 1$. Thus, as *pd* decreases the stimulation effect the anomalous event will have on the detector increases. From the above it is clear that training data regarding the nonself space is required to generate detectors using the hypermutation algorithm.

The second stimulus for a detector is the affinity of the particular detector to the neighbours to which it is connected. This is defined as $ns = \sum_{x=0}^{m} 1 - d_x$, where *ns* is the amount by which the detector is stimulated by its neighbours; *n* is the number of links to the current detector and d_x the distance of the x^{th} neighbour from the detector. This is simply the sums of all the affinity values to a particular detector's *n* neighbours.

A detector is not only stimulated, but also suppressed by loosely connected neighbours. This is taken into account in AIS and is defined by: $nn = -\sum_{x=0}^{n} d_x$ where *nn* is the suppression factor due to its neighbours, d_x represents the distance of the x^{th} neighbour from the current detector (where $0 < d_x < 1$).

The level of stimulation of a given detector is therefore given by: $sl = ps + ns + nn < 1$; where *sl* is the stimulation level for the detector. When a detector's stimulation level exceeds a certain threshold then that detector undergoes cloning.

In the natural immune system, cloning is initiated by a particular B-cell becoming stimulated above a certain threshold. The B-cell replicates, producing large numbers of B-cells and thus antibodies specific to the particular antigen stimulating it. Somatic hypermutation (Farmer et al., 1986) ensures that a relatively large proportion of the clones produced will vary in some way from the parent cell. The cells which remain specific to the original invader are retained within the body and contribute to the immunological memory specific to that particular antigen. Those cells which vary in some way from their parent cell allow the immune system to adapt to variations in the antigen (due to for example a mutation in the infecting agent). This allows the immune system to 'pre-empt' further infections. In the same way, AIS clone and mutate detectors to build up a memory of detectors that can identify similar patterns to the one that caused the cloning.

As stated earlier, the hypermutation method requires knowledge of the nonself space in order to calculate the stimulation level of the detectors. In theory, if the entire self space is known, the nonself space is also known. However, in large systems this is rarely the case, considering the seemingly limitless possibilities for normal process behaviour. With no knowledge of the nonself space, no measure exists as to the performance of the mutated detectors. In this project it is assumed that no knowledge of the nonself space is available which renders the use of the hypermutation method inoperative. It will therefore not be considered in subsequent chapters. The mention of hypermutation is however necessary as it is used in a large number of non process engineering applications.

2.2.5 Applications of AIS

Artificial immune systems have been successfully employed in a wide range of applications (de Castro and Von Zuben, 2000; Garret, 2005). An overview of the broader applications of AIS, not relevant to process engineering is given below, followed by a deeper insight to more related applications.

Forrest et al. (1994) proposed a negative selection algorithm in change detection, which was used to detect malicious code resulting from computer viruses. This led to distributed change detection (Forrest and Hofmeyer, 2000) and to network security (Hofmeyer and Forrest, 2000). Deaton et al. (1997) proposed a means of implementing the negative selection algorithm in a DNA computer. Negative selection is also combined with multiobjective evolutionary programming in a network intrusion detection problem (Anchor et al., 2002). Bradley and Tyrell (2002) used negative selection to build hardware fault-tolerant systems. Although these applications are useful and illustrate the influence of AIS in modern times, they do not fit the nature of this project which is dynamic systems. A problem that closer matches the application at hand is the 'tool breakage detection' problem by Dasgupta and Forrest (1995).

In their paper on 'Tool Breakage Detection', Dasgupta and Forrest (1995) proposed a method based on negative selection that would alert an operator to any changes in the steady-state characteristics of milling cutter dynamics. Their results demonstrated that the proposed algorithm could successfully detect tooth breakage from dynamic variation of cutting force signals. It was reported that the detection system could be quickly updated by generating a new set of detectors as the normal milling operation shifts due to modifications of tool-workpiece geometry, changes in cutting conditions, etc. in the report on tool breakage, Dasgupta and Forrest states that it is possible to combine the data of several sensors (i.e. sensor fusion) in order to improve the reliability of the monitoring system. Variables in their method were window size, matching threshold and number of detectors. The r-contiguous matching rule was used in conjunction with random detector generation. Other matching rules and generation methods were left for future study. The report concluded that the algorithm may be useful for many other similar problems, including fault detection, anomaly detection, machine monitoring, signature verification, noise detection and patient's condition monitoring.

De Castro and Von Zuben (2000) reported an abundance of applications for artificial immune systems, ranging between robotics, optimisation, control, anomaly detection, pattern recognition, computer models and more. However, for dynamic time series data they only proposed one novelty detection algorithm, based on the previously discussed negative selection algorithm. Their algorithm can be summarized as follows:

- Collect sufficient time series data to exhibit the normal behaviour of a system.
- Determine the range of variation of data and perform a binary encoding according to the desired precision.

- Select a suitable window size (concatenation of a fixed number of data points) which captures the regularities of interest.
- Slide the window along the time series, in non-overlapping steps, and store the encoded string for each window as self, from which detectors will be generated.
- Generate a set of detectors that do not match any of the self strings.
- Once a unique set of detectors is generated from the normal database of patterns, it can probabilistically detect any change (or abnormality) in patterns of unseen data.
- While monitoring the system, use the same encoding scheme for the new data patterns. If a detector is activated, a change in behaviour has occurred and an alarm might signal.

These two algorithms are the only two that deal with *dynamic* systems as applied in the field of process engineering. As stated earlier, many applications have proven the efficiency of AIS, but none of them are applied to dynamic process engineering problems.

2.3 Summary of Chapter 2

Chapter 2 commenced by discussing known beliefs regarding the natural immune system and the working thereof. This led to a comparison between the natural and artificial immune systems. Four artificial immune algorithms are discussed in minor detail, these are

- Negative and positive selection
- Clonal selection and hypermutation
- Artificial immune networks
- Danger theory

Of these, the negative selection algorithm is chosen as the preferred method and is therefore discussed in more detail. The real-valued shape space is introduced which is followed by a discussion on the significance of matching rules. Two methods of generating detectors that are commonly found in literature are introduced here, namely random generation and hypermutation.

The chapter concludes by discussing various applications for artificial immune systems as found in the literature.

Chapter 3: Methodology for Process Fault Detection Based on Negative Selection

In this project, the negative selection theorem was chosen for its ability to handle real-valued systems without requiring any knowledge about fault conditions. Different methods for generating detectors are weighed against one another. Different matching rules were also tested.

In order to initiate a successful negative selection algorithm, one needs a dataset (normally a time series) which represents normal behaviour of whatever system is being monitored. This dataset needs to be large enough to include all, or at least most normal behaviour patterns of the process. It should also be accurate and have as little noise as possible.

Once a sufficient dataset has been obtained, the first transformation is made and a matrix (named S) representing the self-space is created. A set of detectors (named R) is now generated from S. At this stage, the artificial immune system has been trained and is ready to monitor for changes or anomalies in the process. As new data values are read from the process, it undergoes preprocessing and is tested against R. If any detector matches the monitored dataset, an alarm is generated. A comprehensive description into this procedure follows.

3.1 Creating the Self-Space

Once historic process data have been accumulated, which is normally a set of time series containing process variables (i.e. temperature, pressure etc.) there is a matter of preprocessing that needs to be attended to. Data are mapped (or embedded) into the shape space by using an adapted embedding method. The first parameter, namely the lag (L) is determined by referring to the autocorrelation function (see Section 3.1.1), while the second, namely the dimension is determined by means of PCA (Section 3.1.2). Space transformations are made after the dimensionality of the system has been determined as discussed below.

3.1.1 Autocorrelation Function

The Autocorrelation function R_L, where L is the specific lag, is defined as the linear correlation of a series of values with the same series with a specified lag (Alciaturi et al., 2005). R_L has values in the interval {-1,1} and is defined as follows:

$$\begin{aligned} R_L &\to -1 \quad \textit{Alternating Series} \\ R_L &\to \ \ 0 \quad \textit{Random Series} \\ R_L &\to +1 \quad \textit{Smooth Series} \end{aligned}$$

Where R_L is calculated as shown in Equation 10:

$$R_L(t) \equiv \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} f(L) f(t+L) dL \qquad [10]$$

When transforming the data into the shape space, one would prefer to choose the lag in such a way that the time series is modelled as a random series. This would ensure that its characterising patterns are fully captured and unchanged. The lag should therefore, be chosen at the lowest value of L where R_L = 0. When combining a number of time series from different sensors to create a self space, as is the case in the three case studies considered, the L for each time series may be kept constant as the L calculated for the first time series. An uncorrelated self space will still be obtained.

3.1.2 Transformation from time-space to shape-space

Once the lag is determined, the next step is to transform the data from the time-space into the shape-space. Consider a set of N time series T_n ($n \ \varepsilon \ \mathbb{N}_0$ {1,..,N}), each with an autocorrelation lag L_n, representing the normal behaviour of a process facility. The self-space S is constructed as follows:

$$\begin{aligned} & S_{n,ij} = T_{n,k} \\ & \textit{where:} \\ & k = (i-1) \times L + j \\ & n \ \textit{ is the number of time series} \\ & S = \begin{bmatrix} S_{1,IJ} & S_{2,IJ} & \cdots & S_{N,IJ} \end{bmatrix} \\ & \textit{where} \\ & I = 1,2,..,L \\ & J = 1,2,..,\min(j_n) \end{aligned} \qquad [11]$$

This means that the time series T_n is divided into windows or packets of a fixed length L, as shown in Figure 12.

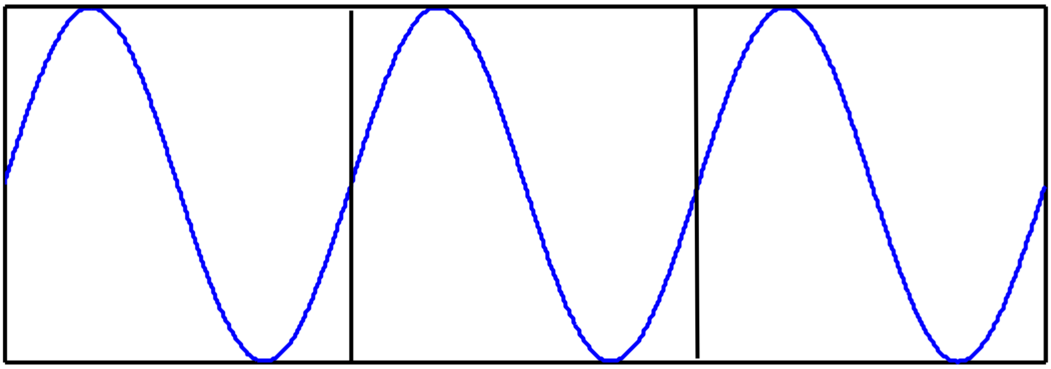


Figure 12: Windows drawn on a time series, indicating the compiling of the self-space.

In Figure 12, let L = 5, S will then be defined as:

$$S = \begin{bmatrix} 0.0 & 1.0 & 0.0 & -1.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & -1.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & -1.0 & 0.0 \end{bmatrix}$$

As shown in Equation 11, the procedure described by Figure 12 is repeated for every process variable T_n. The matrices S_n collectively form the self-space S of size (L x N), describing the interaction of the process variables.

The final matrix S is likely to contain a large amount of redundant information. In order to reduce the dimensionality of this matrix, Principal Component Analysis (PCA) is used (see Chapter 2 for details on PCA). It is essential for the data to be mean centred before applying PCA. Whether the data is scaled or not, depends on the specific scenario. Variance scaling may be appropriate in systems where the variables have different units, e.g. temperature, pressure and concentration. PCA attempts to capture variation. Numerically, a variation in temperature between 1000 and 1100 K is much greater than a variation in concentration between 0.01 and 0.1 M. However, the effect of each of these variables on the system of interest may be very similar and the information content of the temperature data is not inherently greater. For this reason, it may be advisable to autoscale the data. This essentially places all variables on an equal basis in the analysis. In other systems where all of the variables have the same units, e.g. spectroscopy, mean centering may be the appropriate method of choice. Variables with a great deal of variation may be more important, i.e. containing more information about the system under observation, than those with small variation. After applying PCA to the matrix S, the dimensionality of the shape space is determined by considering the Eigenvalues returned by PCA. The number of dimensions that describe approximately 90% of the data variation is chosen as the dimensionality of the shape space.

3.2 Generating Detectors

The overall efficiency of the system will be determined largely on the ability of the detectors to identify process anomalies. There are various methods of generating the detectors, of which three are used in this project, namely: random generation, convex hulls and the hypercube vertex approach. The first two methods are well described in the literature (Dong et al., 2004; Esponda et al., 2004; Fyfe and Jain, 2004; Hunt and Cooke, 1996.). The third method used in this project was developed during the course of the project, namely the *Hypercube Vertex* method. The approach to the detector generation by this method is described in Section 3.2.3.

3.2.1 Random Detectors

Single detectors r are randomly generated so that $p_n \leq r_n \leq q_n$ (n ε $\mathbb{N}_0$ {1,..,N}, where N is the dimensionality of the shape-space); p_n and q_n are the coordinates of the shape-space in the nth dimension. Typical values for p_n and q_n are in the interval {-1,1} for a scaled system. It can also range between say {1.5,1.5} if larger process deviations are expected. These random values are uniformly distributed, as apposed to being normally distributed. This causes the entire shape space to be filled instead of most of the detectors falling in the centre of the shape space.

Once r is generated, the Euclidean distance d_j is calculated for every j from 1 to J, where J is the number of elements in self-set S; i.e. $d_j = \left(\sum_{n=1}^{N} \left(r_n - S_{j,n} \right)^2 \right)^{0.5}$. If the shortest distance between the detector r and its closest neighbouring self elements S_j is larger than some threshold distance ε (epsilon), that is min(d) > ε; r is accepted into the detector set R (determining the size of ε is discussed hereafter). Otherwise, it is rejected and a new r is generated randomly. This procedure is repeated until R has reached a given size (to be discussed later). The routine can be accelerated by reducing the size of the search space to a given hypercube with centre coordinates r, instead of the entire shape-space. The size of d_j is then drastically reduced, which lessens the computational strain. The size of this hypercube should be at least twice the length of ε {r_x + ε ; r_x – ε} to avoid misclassification of possible detectors.

The threshold ε, is also used in monitoring the process, which is again referred to in subsequent chapters. The value for ε is estimated prior to the generation of detectors, by calculating the maximum distance between two neighbouring points in the self space, i.e. $\max(d_{ij}) = \max\left(\sum_{n=1}^{N} \left(S_{i,n} - S_{j,n} \right)^2 \right)^{0.5}$ where i ≠ j. Once this figure has been calculated, ε is estimated as half an order of magnitude (5) greater than this distance. This value was determined empirically and it ensures that the detectors are neither under- nor over-fitted. This value is dependant on the sampling rate of the system, and varies between case studies as mentioned in Chapter 5.

Another parameter under the Random Generation mechanism is the minimum number of detectors necessary for efficient fault detection. This figure can be calculated by considering the so-called detector coverage. This method estimates the percentage of the nonself space that is covered by the current number of detectors, given the current affinity threshold. The percentage detector coverage is estimated by regarding the number of detectors that overlap as a percentage of the total number of detectors. If 100% of the randomly generated detectors in the shape space overlap, i.e. $\min(d_{ij}) = \min\left(\sum_{n=1}^{N}(r_{i,n} - r_{j,n})^2\right)^{0.5} < \varepsilon$, where i ≠ j, it can be assumed that ~ 100% of the detectable nonself space is covered, obviously excluding holes.

3.2.2 Convex Hulls

When the shape-space is inspected, in any number of dimensions, a boundary can be drawn between the self-space (S) and the nonself-space (M). Points on this boundary then define the detector set (R) and isolate the self-space. In order to successfully isolate the self-space, one needs to consider the holes within the self-space. In order to generate detectors on the boundaries that demarcate holes in the self space, one has to remove the outer boundary points of the self-space one layer at a time, until only the inner-most layer remains.

Using this principle, detectors can be generated by identifying the outer and inner boundaries of an attractor. It may be necessary to first add noise to the system, in order to prevent generating a detector that belongs to the self-space. Every time random noise is added to the system, a different set of boundary detectors are obtained, therefore the procedure may be repeated until a given number of detectors are collected. Equation 12 explains this concept:

$$
\begin{aligned}
&\hat{S} = S \cdot \varepsilon \cdot rand(size(S)) \\
&R = S(unique(convhull(\hat{S}))) \\
&where \qquad\qquad\qquad\qquad\qquad\qquad \text{[12 a-b]} \\
&\varepsilon \;\; is \;\; the \;\; affinity \;\; threshold \\
&convhull \;\; is \;\; a \;\; MATLAB \;\; command \;\; for \;\; calculating \;\; convex \;\; hulls
\end{aligned}
$$

Figure 13 shows a set of detectors generated using the convex hulls approach.

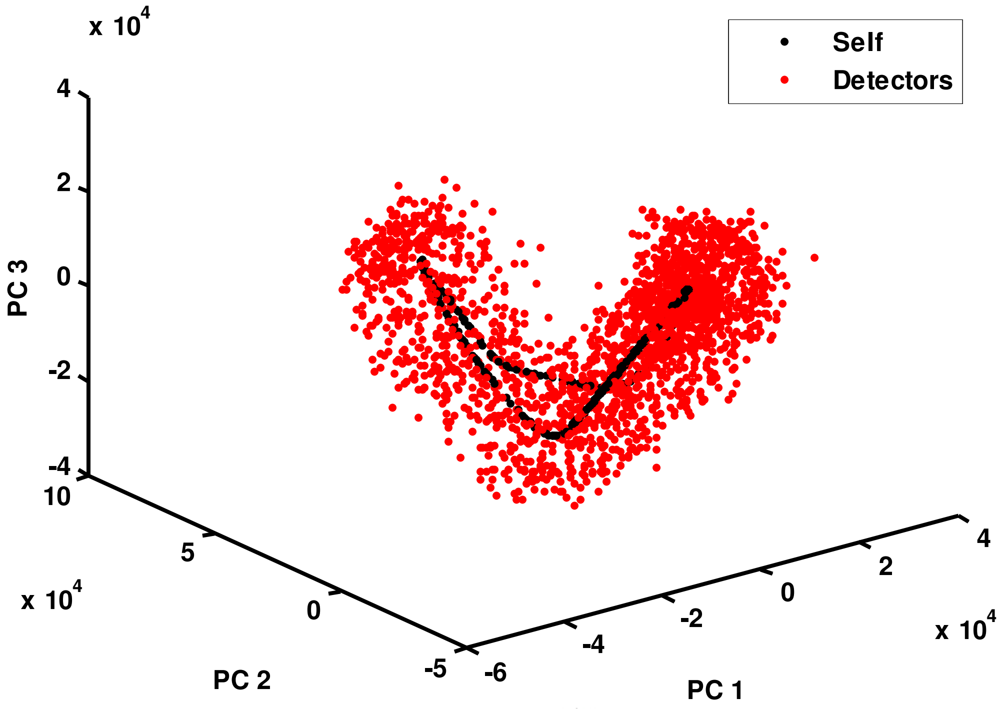


Figure 13: Detectors generated in a 3-dimensional Shape-Space using Convex Hulls

3.2.3 The Hypercube Vertex Approach

Although the abovementioned methods are well defined in the literature (Esponda et al., 2004; Dong et al., 2006), they each seem to have their drawbacks (Garret, 2005). Industrial processes have scores of variables and require a large process history to explain all normal events. Random generation is extremely computationally demanding in large systems. The time needed to generate the detector set increases exponentially as the required size of the set increases (Esponda et al., 2004). Recent literature claims that Negative Selection is reserved for smaller or binary systems (Hunt and Cooke, 1996; Dong et al., 2006). In a binary alphabet, using the hamming distance matching-rule, random detectors are generated fast and respond well (King et al., 2001). The Negative Selection Algorithm is highly regarded in network intrusion and other such binary systems (Esponda et al., 2004). In real valued systems, problems start to arise regarding computational cost and detector efficiency (Dong et al., 2006). As the size of the system escalates, the impact of these issues escalates. With the development of the Hypercube Vertex method, these concerns are hoped to be eliminated.

The Hypercube Vertex approach positions 2xN detectors r_i, where N is the dimensionality of the shape-space, around points in the self space:

$$\begin{aligned} & r_i = S_{i,n} \pm \delta \\ & where \\ & i = 1,2,.....,length(S) \\ & n = 1,2,....,N \end{aligned} \qquad [13]$$

The parameter δ is introduced to position the detectors at a given distance from the self elements S_i. The magnitude of δ is determined by (as is the case with ε) considering the distance between neighbouring points in S, i.e. S_i and S_{i+k}. k is not necessarily = 1, since a second trajectory can place a point between S_j and S_{j+1}. The Euclidean distance matrix E_{ij} is calculated between all self elements S_i and S_j, where both i and j runs from 1 to J and J is the number of elements in self-set S; i.e. $E_{ij} = \left(\sum_{n=1}^{N} \left(S_{i,n} - S_{j,n} \right)^2 \right)^{0.5}$. The smallest $E_{ij} \neq 0$ is obtained and converted by a constant c to produce: $\delta = \min\left(E_{ij} \times c \right)$, where c was empirically determined to be 6, but is not restricted to these values. This will result in δ being 20% larger than ε. The parameter δ is introduced to accommodate noise in the system, as it adds necessary holes, as described in section 2.2.3c. The use of δ is not restricted to systems with noise; it also provides a margin of error for systems in which the self space is not 100% defined. The less noise present in a system and with an increased representation of the self space, the required value of δ will drift closer to the value of ε, until finally they merge.

Detectors need not be generated around every point in the self space since they tend to fall far closer together than the sensitivity radius ε. Every nth point is thus used in detector generation. It is recommended that n be a prime number between 1 and 13 (depending on the required downscaling of the number of detectors), since this maximises the chance that most sections on the attractor is covered without adding unnecessary detectors. By varying the number of detectors, a better comparison can be made between the different methods of generating detectors.

Inappropriate detectors are hereafter removed in the same way that random detectors are verified: an Euclidean distance matrix d_{ij} is calculated between every detector r_i for i from 1 to I, where I is the number of detectors in R and every self element S_j, where j runs from 1 to J and J is the number of elements in self-set S; i.e. $d_{ij} = \left(\sum_{n=1}^{N} \left(r_i - S_{j,n} \right)^2 \right)^{0.5}$. r_i is accepted into the detector set R only if the shortest distance between the detector r_i and its closest neighbouring self elements S_j is larger than some threshold distance ε (epsilon), that is min(d_{ij}) > ε.

Informally, Figure 14 represents an enlarged trajectory of a system's attractor. For simplicity, a two dimensional attractor is considered.

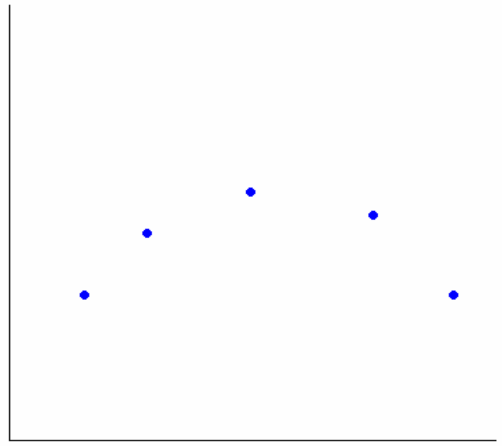


Figure 14: Enlarged Trajectory of a system's attractor

The goal is to add detectors in close proximity of this trajectory, ensuring that no detector falls within the trajectory, or close to a point in the trajectory. This is achieved by adding an arrangement of detectors around each point in the trajectory and then removing all detectors that do not comply with the above restraints. The outcome of this method is shown in Figure 15, where the blue dots represent the trajectory and the red dots represent the detectors. A set of four (2 x number of dimensions) detectors is scattered around every point in the attractor. Detectors that fall too close to or inside the attractor's trajectory are then removed. This is similar to the process that occurs in the human thymus, where receptor bearing T-cells have which react with known proteins, are destroyed.

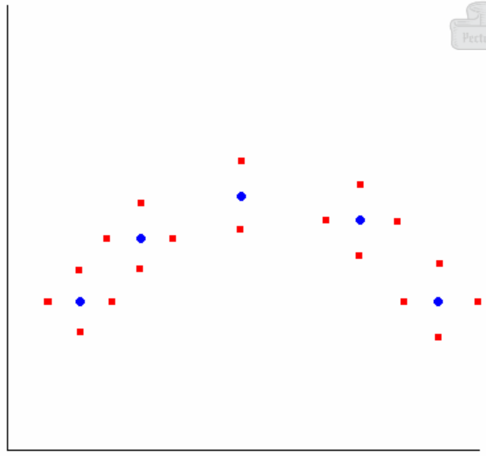


Figure 15: Detectors added to a system's trajectory

The method in placing detectors around a specific point relies on a simplistic mathematical approach. For a two dimensional attractor, every point will have a [x,y] coordinate system. The detectors are created by adding a distance (δ) to each of the coordinates respectively. The coordinates of the four detectors surrounding a given point on the attractor will therefore be:

- [x + δ , y]
- [x - δ , y]
- [x , y + δ]
- [x , y - δ]

This can also be done in higher dimensions and the number of detectors surrounding a given point will always be 2 x n (n = number of dimensions).

Any detector will therefore, after removing illicit detectors, not be closer than δ units from the attractor. This can be proved using the equation for calculating the distance between two points on a Cartesian axis:

$$\begin{aligned} dist &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ &= \sqrt{(x + \delta - x)^2 + (y - y)^2} \\ &= \sqrt{\delta^2} \\ &= \delta \end{aligned} \tag{14}$$

Figure 16 illustrates the manner in which detectors are added to the system. Note the structure in which they are added, as opposed to the unordered placement in the previous methods.

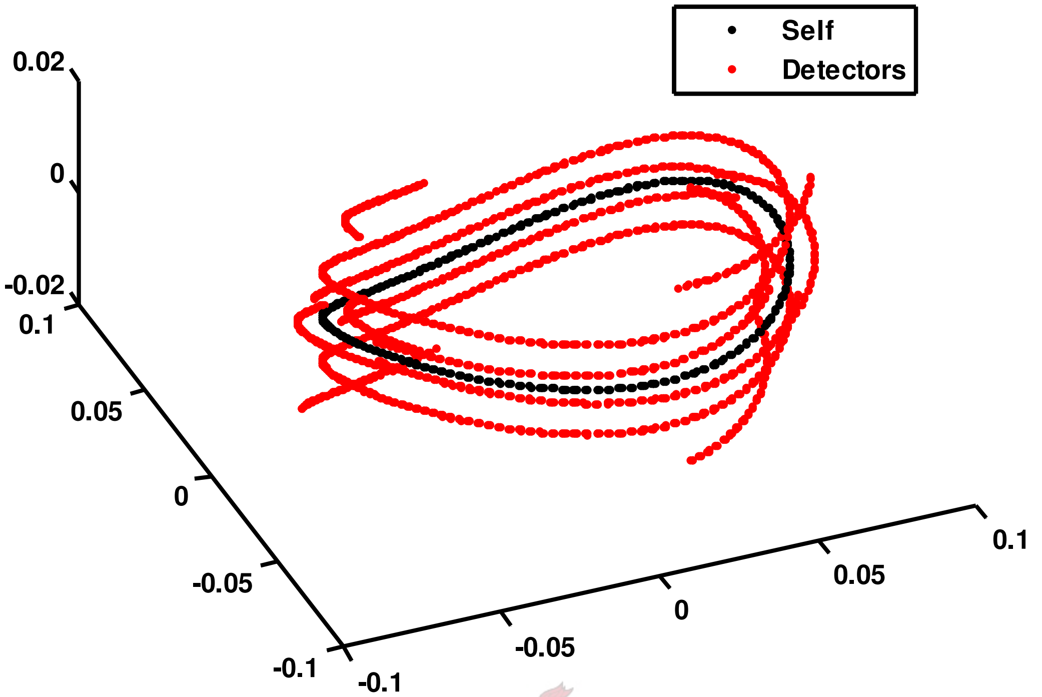


Figure 16: Detectors generated on the hypercube vertices of the self-events

3.2.4 Computational Cost of the various methods

There are two factors that influence the efficiency of a given method for generating detectors. The first is probably the most important, which is the ability of the generated detectors to identify a process anomaly, without false alarms as described by the ROC analysis. The second is the computational cost in generating these detectors. Although 21st century resources soften the impact of computational stress, the matter still requires a certain degree of consideration.

The relative computational cost for generating detectors is tested by comparing the time required to generate a given number of detectors. Each method is then used to generate a fixed number of detectors. These experiments are all conducted on the same computer, where the processing time for each run is timed and compared.

3.3 Monitoring the Process for which the AIS was trained

Once a matrix S containing “self” and a matrix R of detectors have been created, one can start monitoring process data T_{new} in real time. The same matching rule is used to monitor incoming data as was used for testing detectors against the “self”.

In real time, data are gathered one data point at a time. Data points are accumulated until a window or packet is filled, at which stage the packet undergoes the same preprocessing as the original time series data T, i.e.

$$\begin{aligned}
&M_{n,ij} = T_{NEW\,n,k} \\
&where: \\
&k = (i-1) \times L + j \\
&n \;\; is \;\; the \;\; number \;\; of \;\; time \;\; series \\
\\
&M = \begin{bmatrix} M_{1,IJ} & M_{2,IJ} & M_{N,IJ} \end{bmatrix} \\
&where \\
&I = 1,2,..,L \\
&J = 1,2,..,\min(j_n)
\end{aligned} \qquad [15\ \text{a-b}]$$

As soon as one window has been filled, it is tested for a match with any of the detectors r_i in the matrix of detectors R, by calculating the Euclidean distance d_i for every j from 1 to J, where J is the number of elements in R; i.e. $d_j = \left(\sum_{n=1}^{N} (M_n - r_{j,n})^2 \right)^{0.5}$. If the shortest distance between the closest detector r_j and the specific element M is less than the threshold ε, i.e. min(d) < ε; a match is made and an alarm is triggered, also referred to as an immune response. The index of the triggering detector (j) may or may not be noted for diagnostic purposes.

3.4 Efficiency rating of a classifier by means of ROC Analysis

ROC (Received Operating Characteristic) analysis is generally used in signal detection theory (Stibor, 2005). It provides a measure of a classifier's performance by taking into consideration the trade-off between hit rates and false alarm rates. Consider a process with 100 events in the shape-space, of which 99 are non-self and 1 is self. Now consider a classifier that always predicts non-self. In this case, the classifier's error rate is 1% and is consequently fairly accurate. If this classifier is however used on a different dataset, of which 99 is self and 1 non-self, the classifier error rate is 99% and is inadequate. This problem occurs because the class distribution is unknown. As shown in Table 2, there are four different categories into which an event can be classified. In order to determine the true efficiency of a classifier, they need all be taken into consideration. The trivial classifier mentioned above has a detection rate of 100%, but a false alarm rate of also 100%, which deems it ineffectual. To compare the classification performance of classifiers using the ROC analysis, the detection rate and false alarm rate is plotted on a two dimensional graph, known as the ROC-space. The detection rate is plotted on the ordinate and the false alarm rate on the abscissa. This means that the classifier at point (0,1) is a perfect classifier (100% detection with 0% false alarms).

Table 2: Classification types (Stibor et al., 2005).

Event	**Classified as Normal**	**Classified as Anomalous**
Normal	True Negative	False Positive
Anomalous	False Negative	True Positive

The ROC analysis can be used to find the optimum detector radius r_d, by assigning a small radius at first (e.g. 0.01) and increasing this radius by a small value ($r_d = r_d + \Delta r$) until r_d reaches some predefined maximum (e.g. 1.0). The detectors' rating under ROC analysis is plotted in the ROC-space to form the so-called ROC-curve. An optimum radius can then be chosen which yields the minimum error.

3.5 Summary of Chapter 3

The various transformations and calculations needed in creating AIS were discussed in this chapter. Section 3.1 introduced the autocorrelation function, which is used to determine the dimensionality of the shape-space. A description of the transformation between the measurement space and shape space followed, which led to a description of the various mechanisms for generating detectors. These mechanisms were formally and informally explained while comparisons thereof are provided in the next chapter. Mention was made to the matter of computational cost regarding the generation of detectors. This chapter concludes by discussing monitoring process behaviour, using trained AIS.

The general procedure of creating, training and using AIS are summarised in Figure 17.

Three parameters are involved in the creation of an artificial immune system, they are:

- Autocorrelation Lag, L
- Number of detectors, n.
- Detector affinity threshold, ε.

L is determined by considering the autocorrelation function R_L for the given process. L is chosen as the smallest value for L such that $R_L = 0$. The number of detectors (n) is determined by estimating the detector coverage. The affinity threshold ε is calculated as 5 times the maximum distance between neighbouring events in the self space, while δ (a fourth parameter only used in the hypercube vertex approach) is calculated as 6 times this distance. It is important to realise that none of these parameters are calculated by considering the nonself space. No knowledge is available of the nonself space when creating the AIS. In the Chapters 4-6, the nonself space is considered merely as a test to validate the capabilities of the various systems.

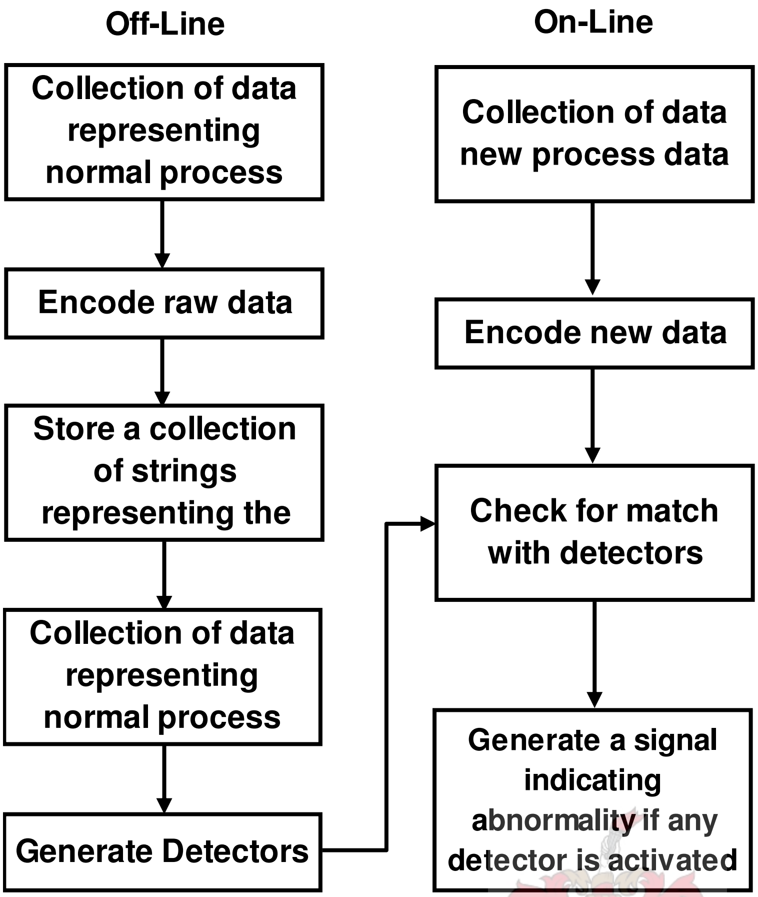

Off-Line
On-Line
Collection of data representing normal process
Encode raw data
Store a collection of strings representing the
Collection of data representing normal process
Generate Detectors
Collection of data new process data
Encode new data
Check for match with detectors
Generate a signal indicating abnormality if any detector is activated

Figure 17: Schematic diagram showing the various stages of processing for fault diagnosis with the artificial immune system (Dasgupta, 1999).

To summarise the methodology regarding the negative selection algorithm, consider a time series T. The following procedure will create a negative selection algorithm to monitor a dynamic system for deviations from normal process behaviour.

- Use Equation 10 to determine the autocorrelation function of the time series. If more than one time series is present, the lag as determined for the first time series is sufficient for use with all the series.
- Use Equation 11 to transform the time series into a preliminary self matrix S.
- Reduce the dimensionality (if needed) of the preliminary self matrix S by means of PCA. The final dimensionality of the shape space is determined by considering the eigenvalues returned by PCA. A standard is set at 90% of the data variation, which should be covered by the number of dimensions decided upon.
- Determine the detector affinity threshold by calculating the maximum distance between neighbouring events in the self space and multiplying it by 5 (this value was determined empirically).
- Generate detectors by using any of the three methods described:
    - Random Generation: Generate detectors randomly and remove those in whose affinity threshold lays a self event. Two parameters are present in this algorithm, i.e. the detector affinity (ε) and the number of detectors (N). The detector affinity is estimated by calculating the

maximum Euclidean distance between neighbouring events in the self space and multiplying this figure by 5 (empirically determined). The number of detectors is hereafter determined by estimating the detectors' coverage of the nonself space. If 90% of the nonself space is covered, the required number of detectors has been reached.

  - Convex Hulls: Add random, uniformly distributed noise to the self space and calculate the convex hulls of the noisy self set by means of Equation 12. This procedure identifies points on the self space that become detectors. This procedure is repeated until the required number of detectors is reached. Again, two parameters exist in the method, the detector affinity (ε) and the number of detectors (N). These parameters are determined in the same way as for random generation.
  - Hypercube Vertex: This method places 2x*D* detectors around every *n*th point in the self space, where *D* is the dimensionality of the self space as determined by PCA and *n* is a prime number that adjusts the number of detectors (N). Again, detectors that are positioned such that a self event falls within the boundaries of their affinity thresholds are removed. Apart from the above, two more parameters exist in this method; they are the affinity threshold, ε (calculated as 1.67 times larger than ε in the previous two methods) and δ which is constant at 1.2 x ε. Both of these values were determined empirically.

- Monitor the process at hand by testing for a match between the current event and all available detectors. A positive match will result in an anomaly being detected. The Euclidean distance rule is once again used for the test for matching events/detectors.

Chapter 4: Applying AIS: The Volterra Process (Predator-Prey System)

4.1 Introduction

The first case study being considered is a Lokta-Volterra process, more specifically, the Predator-Prey System. This process is based on the natural equilibrium in a 2-species symbiosis (Weisstein, 2003). It assumes that a given predator has only one prey and vice-versa. Further, it assumes that predators only die of natural causes. The Predator-Prey system is described by two linear co-dependent differential equations, shown in Equation 16 (a and b), where x_1 and x_2 are the number of prey and predators respectively.

$$\frac{dx_1}{dt} = k_1 \cdot x_1 - C \cdot x_1 \cdot x_2$$

$$\frac{dx_2}{dt} = -k_2 \cdot x_2 + D \cdot x_1 \cdot x_2$$

$$\begin{aligned} &Where: \\ &k_1 = 2 \\ &k_2 = 10 \\ &D = 0.002 \\ &C = 0.001 \end{aligned} \qquad [16\text{ a-b}]$$

In these equations, each variable represents a different dynamic attribute of the system. They are:

k_1 : Prey birth rate (Prey•time^{-1})

k_2 : Predator death rate (Predator•time^{-1})

C : Kill rate (Prey•time^{-1})

D : Predator surplus (Predator)

When these constants are varied or allowed to drift, so will the dynamics of the process, which should be revealed when the process is monitored by a fault diagnostic system. In order to test the capabilities of AIS, two variables are allowed to drift, namely C and D.

This variable drift is commenced after a stage of steady state operation is achieved. At the resulting second steady state, these variables have doubled in value. Three datasets are captured (each being 5000 observations in size), namely: the first steady state (s), the transition phase (m1) and the second steady state (m2). A fixed sampling rate of 0.1 is used throughout. The equations are solved by using the

Runge-Kutta method.

Figure 18 shows a representation of the Predator-Prey equilibrium under normal conditions (s) and the second steady state (m2). As the number of predators increase, the number of prey decreases, due to a high killing rate. This eventually causes the number of predators to decrease, due to a lack of sustenance. This is a not uncommon occurrence in nature and the dynamics are described quite accurately by the two differential equations above.

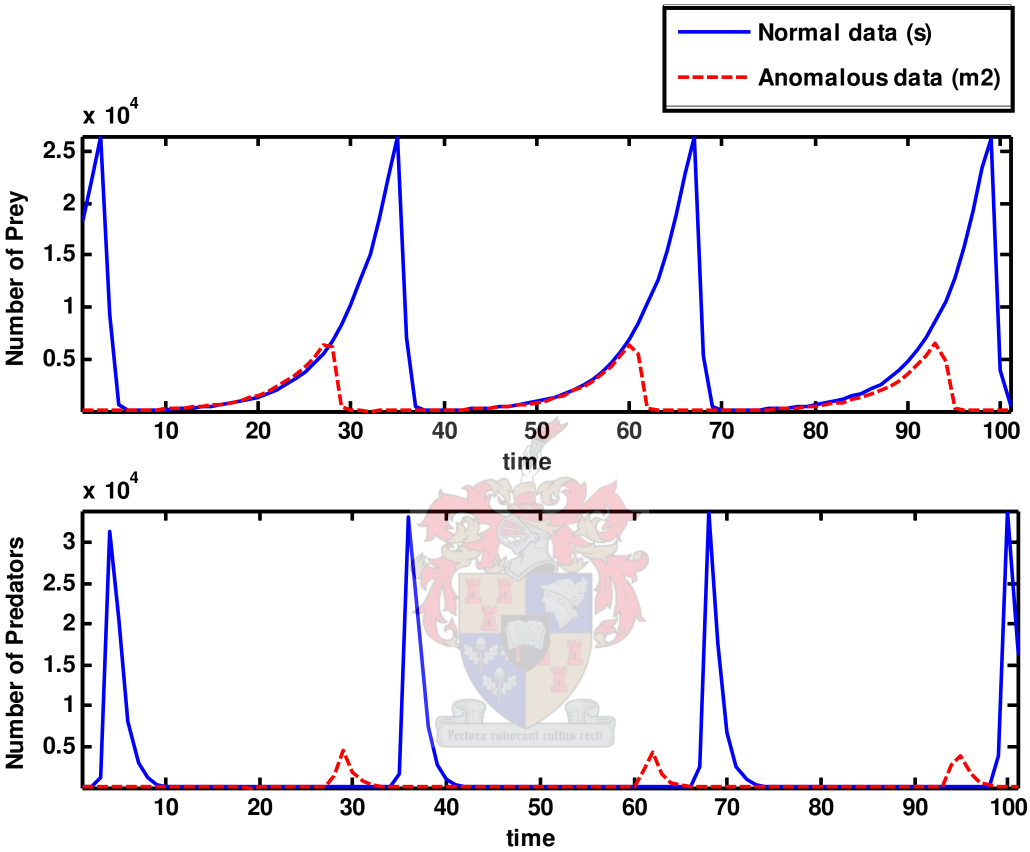


Figure 18: Representation of the Predator-Prey equilibrium: The first steady state is indicated in a solid line and represents normal process behaviour; the second steady state is shown in a dotted line and represents anomalous behaviour.

The only visual difference between the two steady states in Figure 18 is the maximum number of animals in the habitat. Under the altered conditions, fewer animals inhabit the specific control area. From Figure 18, it appears as if the predators become extinct every so often. In theory, predator-prey pairs with extinction-prone local populations can persist through metapopulation dynamics, wherein local populations fluctuate asynchronously, occasionally providing dispersers that prevent permanent extinction in all patches (Holyoak and Lawlak, 1996). The extinction of the predators in this project is no cause of concern, because it is simply a simulation of an arbitrary system. Assigning different values for the model parameters will yield different dynamics.

4.2 Transforming between measurement space and shape space

In order to transform the time series data, s (measurement space) into the shape space S, the autocorrelation lag of this time series needs to be determined. Figure 19 shows the autocorrelation function for the time series data (s).

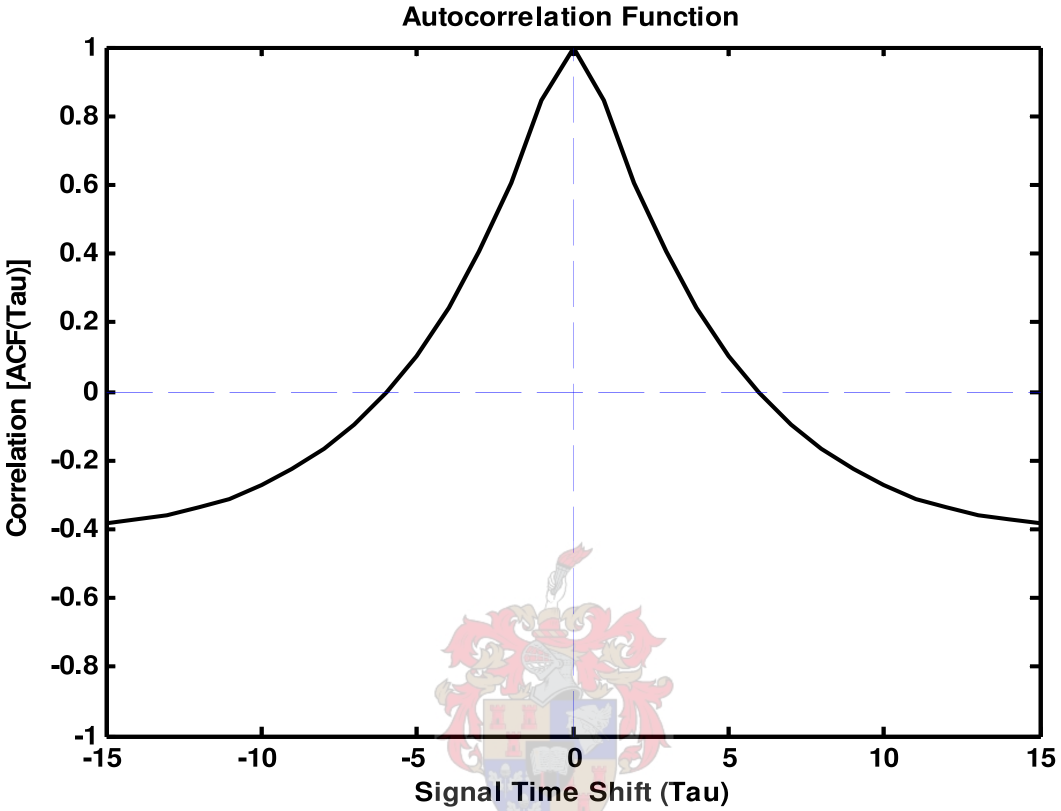


Figure 19: Autocorrelation function of time series data (s) for the Predator-Prey data.

From Figure 19 it can be seen that a time-shift (lag) of 6 measurements provides an uncorrelated transformation. The time series describing the population of the prey is used to determine this lag.

The data are scaled and mean centred, hence the transformation is commenced as described in Chapter 3, with a lag of 6. PCA is applied to reduce the dimensionality of the shape space in order to escape the curse of dimensionality. The shape space is reduced to 3 dimensions, which is illustrated by Figure 20. The cumulative latency of the data suggests that 87.1% of the data variation can be retained by using 3 dimensions in the shape space. There is no predetermined criterion for deciding on the number of dimensions, but one would prefer to capture around 90% of the data with a reasonable number of dimensions.

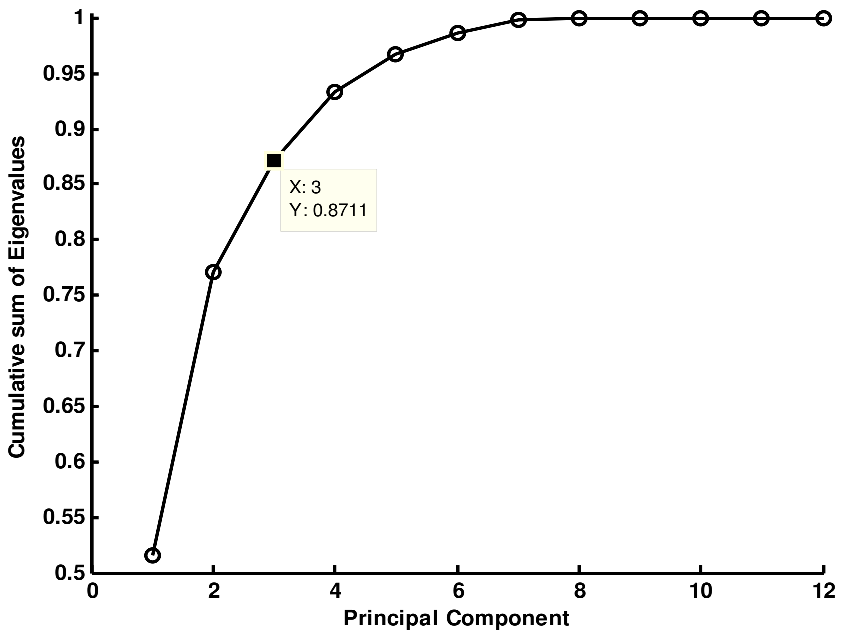


Figure 20: Cumulative sum of latencies of the principal components of self space S of the Predator-Prey data.

The question as to why a seemingly 2-dimensional problem (predator vs. prey), is now transformed into a 3-dimensional shape space might now be raised. Once a dynamic change is introduced (drifting model parameters), a third dimension is required to accurately describe the shape-space, i.e. the shift has occurred in the original plane. A three dimensional attractor is then constructed to represent the self space, by using the first three principal components as calculated by PCA. Refer to Figure 21.

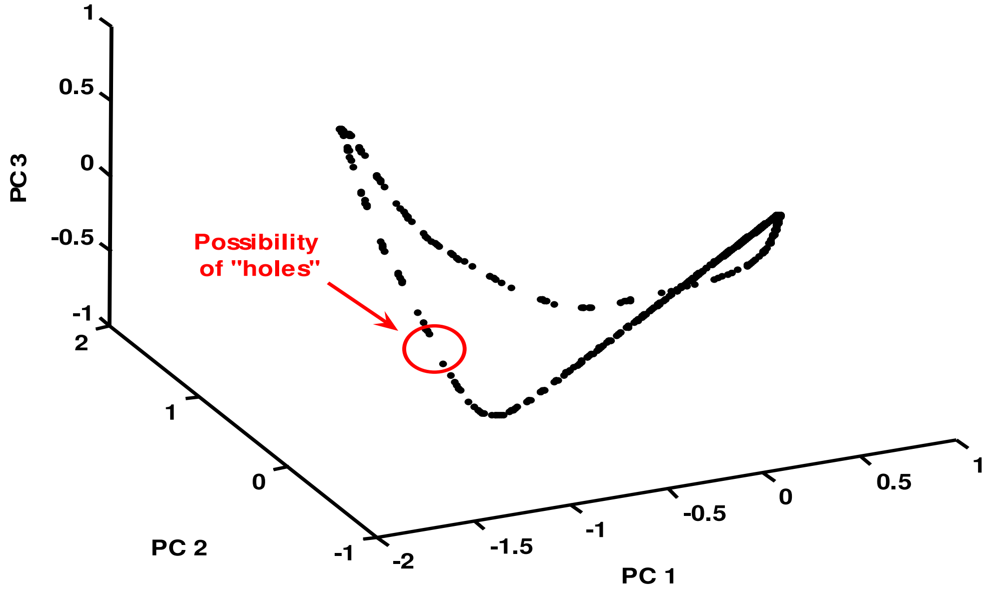


Figure 21: 3-Dimensional Principal Components plot of the Self-Space of the Predator-Prey data

A number of possible holes are identified between points in the trajectory of Figure 21, since there are gaps between neighbouring events that are much larger than average. One of these gaps is marked with a circle.

4.3 Generating Detectors and Classifying Data

Once the shape-space is determined and the self-space is constructed, the generation of detectors may be initiated. Three different methods are tested and are discussed below. In all three methods, the detector affinity threshold ε is varied after being estimated by considering the maximum distance between neighbouring points in the self space. This variation in ε is done simply for validation purposes. In real applications of AIS, no knowledge of the nonself space is available prior to the monitoring stage.

4.3.1 Randomly Generated Detectors

When generating detectors randomly, there are two variables to consider, they are:

- The number of detectors, N
- The size of the affinity threshold, ε.

As discussed in Chapter 3, the detector coverage can be estimated by the percentage of detectors that overlap. Figure 22 shows such an estimation for different numbers of detectors. This curve is not necessarily constant for different sets of detectors, since they are generated randomly and its arbitrary nature is passed on to the estimated detector coverage. The basic curve does, however, remain constant.

Before deciding on the number of detectors, the detector affinity threshold needs to be determined. This is done by calculating the maximum distance between two neighbouring points in the self space, as discussed in Chapter 3. This distance is then multiplied by half an order of magnitude (5), to prevent over- or under-fitting. In this case study, ε was found to exist close to the 0.4 mark. To validate this, a number of tests are performed which alters ε (and in turn also alters the number of detectors) to four different sizes. The results are shown in Table 3.

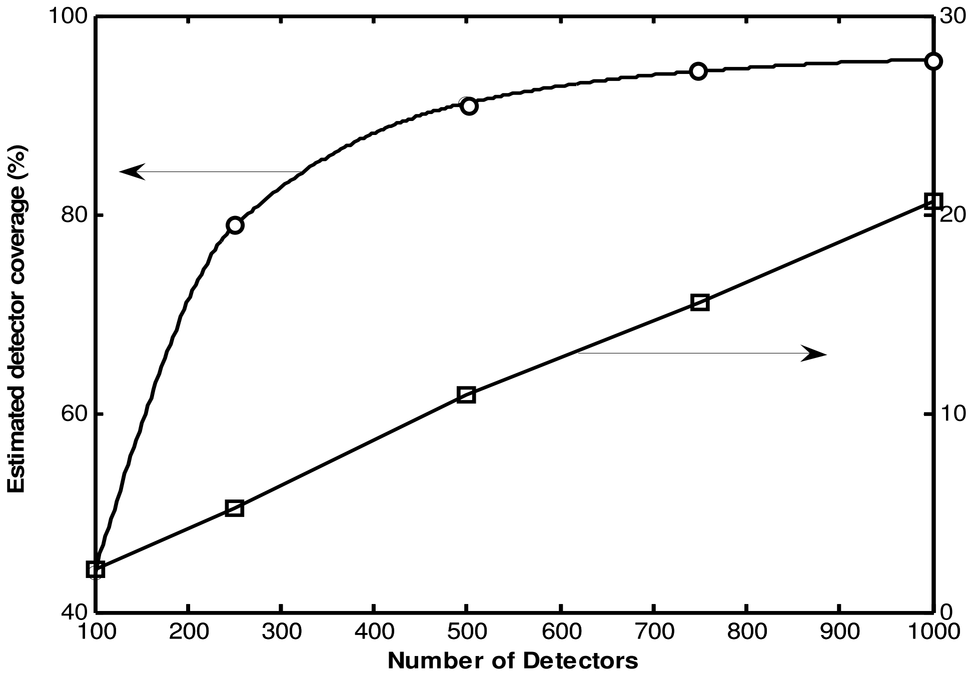


Figure 22: Estimated random detector coverage and time taken to generate these detectors for the predator-prey sistem as a function of the number of detectors. The parameter ε is constant at its predetermined value of 0.4.

For every given value of ε, the number of detectors is chosen by calculating the estimated detector coverage. For ε=0.4, Figure 22 shows that 100 detectors cover approx. 44% of the nonself space, while 1000 detectors cover approx. 95%. Initially, the detector coverage increases rapidly, thereafter it reaches an asymptote and no significant increase in coverage is seen for a large increase in the number of detectors. The time needed to generate x detectors increases linearly with respect to x. The major change in rate of coverage increase is reached at or around the 500 detectors mark. It is therefore decided that a detector set of 500 detectors is an acceptable approximation to the optimal number of detectors. This procedure is repeated and the numbers of detectors are tabulated in Table 3, along with their respective values for ε.

Figure 23 shows the shape space with 500 random detectors, drifting data (s1) and the alarms generated on certain data points, while the performance of the AIS is assessed and summarised in Table 3

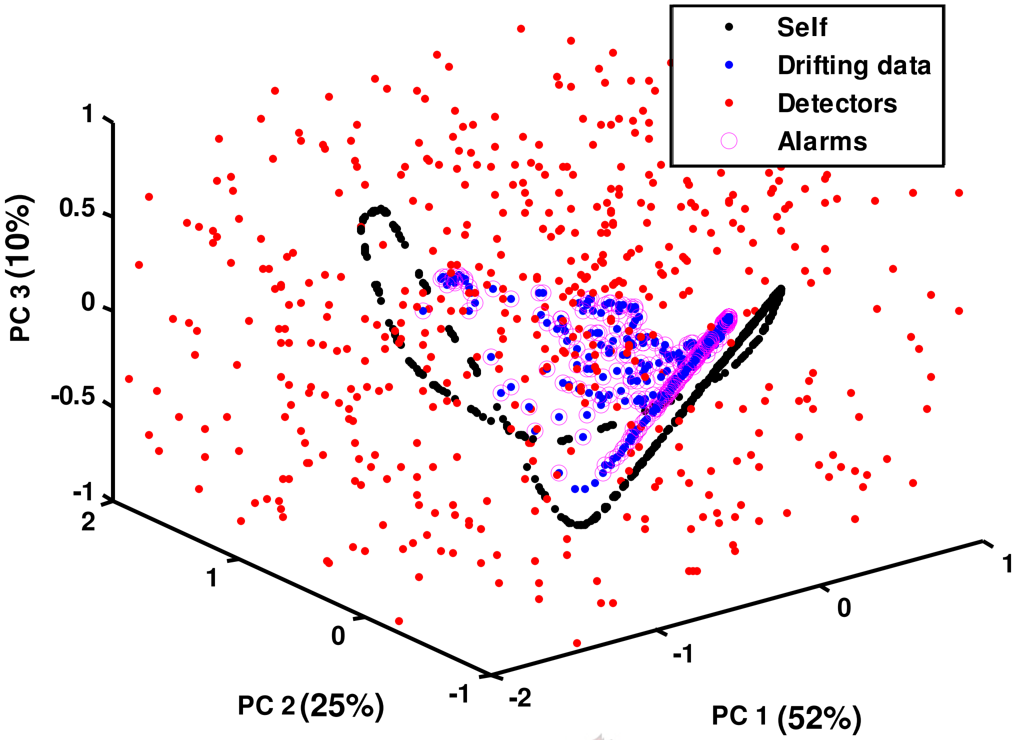


Figure 23: 3-Dimensional presentation of 500 random detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Table 3: Summary of AIS detection capability for the Predator-Prey data, monitored by 500 randomly generated detectors.

Run	ε	Number of detectors	Alarms Triggered		
			Validation (s)	Drifting (m1)	Drifted (m2)
1	0.2	2000	0.7%	92.9%	99.9%
2	0.3	900	0.0%	87.9%	90.8%
3	*0.4*	*500*	*0.7%*	*89.6%*	*92.8%*
4	0.5	300	0.0%	79.9%	82.3%

Table 3 verifies that ε = 0.4 (the row in *italic* print) performs best in monitoring anomalous data. It identified a large percentage of anomalous date (second only to ε = 0.2) without adding excessive computational strain (as is the case with ε = 0.2). it could therefore be said to be the preferred configuration for the randomly generated detectors. It should be noted that the values given in Table 3 is by not absolutely constant and might change to a small extent every time a new set of random detectors is generated. Various detector sets' abilities to detect change does however remain similar (given a constant ε and sufficient coverage), according to experimental results. If too few detectors are chosen, however, (say for instance 200 detectors in this case, with a relatively small ε) the percentage alarms generated between two sets of detectors will show a stronger random tendency. No particular value for ε would, in this case, be noticeably better than any other. Figure 24 shows the ROC-representation of the various different values for ε, with 500 detectors. The originally estimated value of ε = 0.4 is clearly the highest scoring quantity.

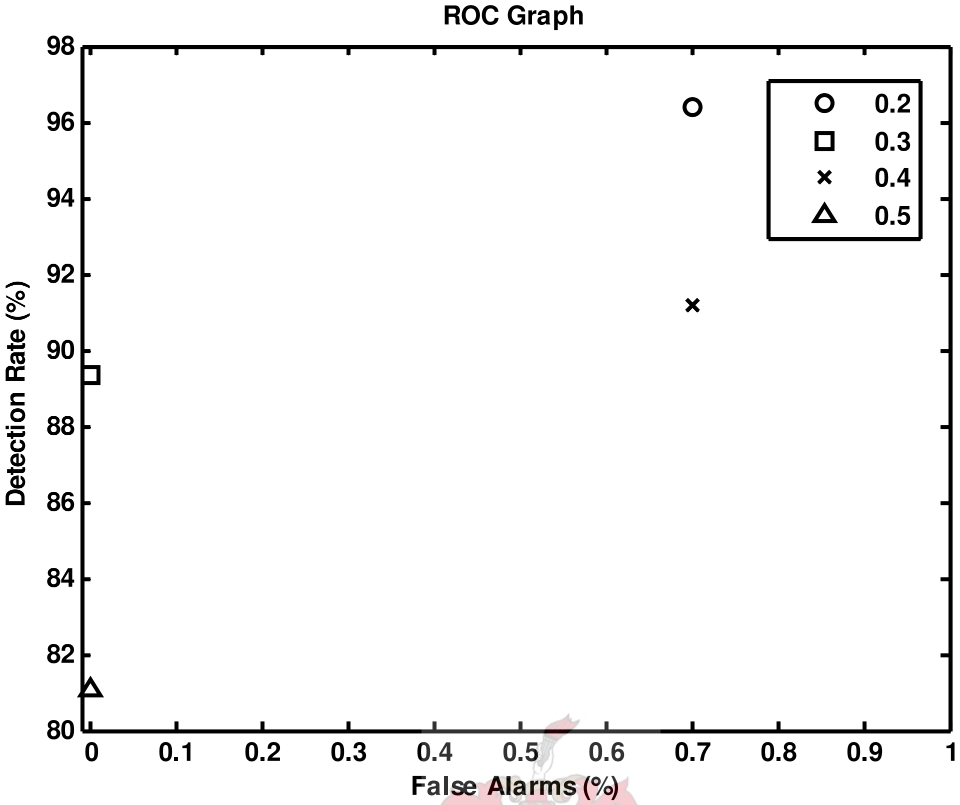


Figure 24: The ROC graph, comparing various different values for ε during random generation.

4.3.2 Convex Hulls

This method uses the theory of convex hulls to identify points on the outer perimeter of the self space and convert them to detectors. To avoid autoimmunity, uniformly distributed noise is added to the self space before generating detectors. This process is repeated, with different noise until a given number of detectors are generated. The number of detectors is once again estimated by considering the detector coverage and number of overlapping detectors. It is found that 100% of the detectors overlap more rapidly than the random detectors. This is because each detector is arranged at a random distance around the self space, which causes the detectors to fall closer together and closer to the self space. Outliers are rarely found and the far corners of the nonself space are not covered. The estimated detector coverage is therefore misleading in this case, but still holds, since anomalous events cannot fall in any uncovered region of the nonself space without first passing through the layer of detectors. It is also found that the time taken to generate any amount of detectors remains relatively constant, e.g. it takes 0.07 seconds to generate 50 detectors, while 500 detectors are generated in 0.09 seconds. These results are displayed in Figure 25.

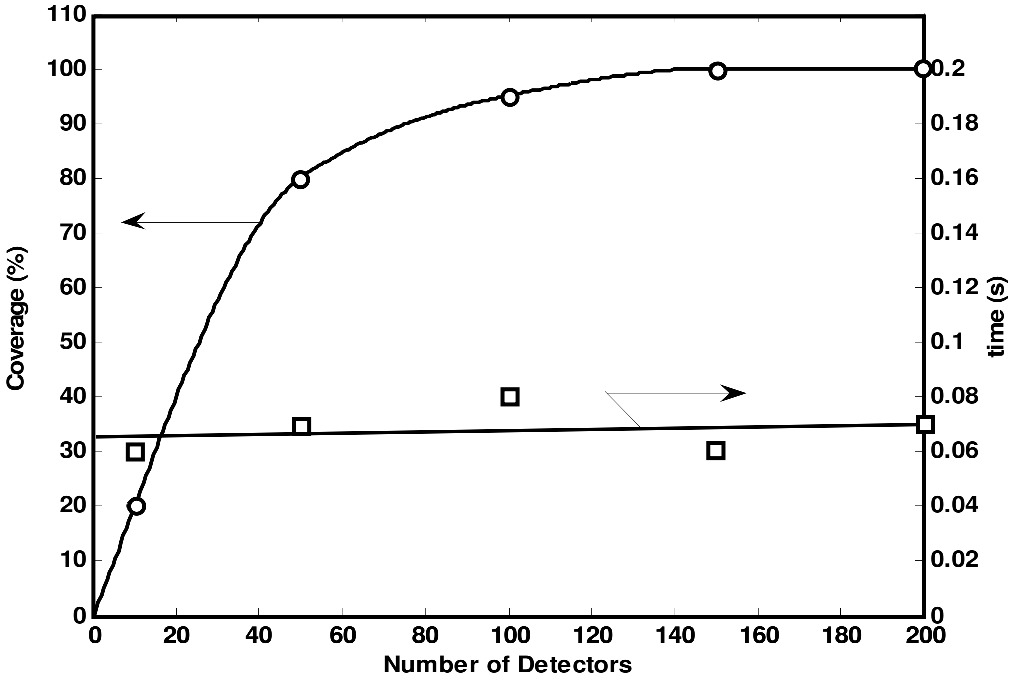


Figure 25: Estimated convex hull detector coverage and time taken to generate these detectors as a function of the number of detectors. The parameter ε is constant at 0.4.

For simplicity, the number of detectors is kept constant in this section. Experiments indicate that choosing the number of detectors at 500 results in an estimated coverage near 100%. This is true for smaller values of ε such as 0.2, while larger values cover 100% of the nonself space with fewer detectors. Because no notable computational strain is added by adding more detectors, nothing is lost in keeping the number of detectors constant at 500.

Once the detectors have been generated, anomalous data can be classified. The shape space is shown in Figure 26, along with anomalous data of which the true positives are encircled.

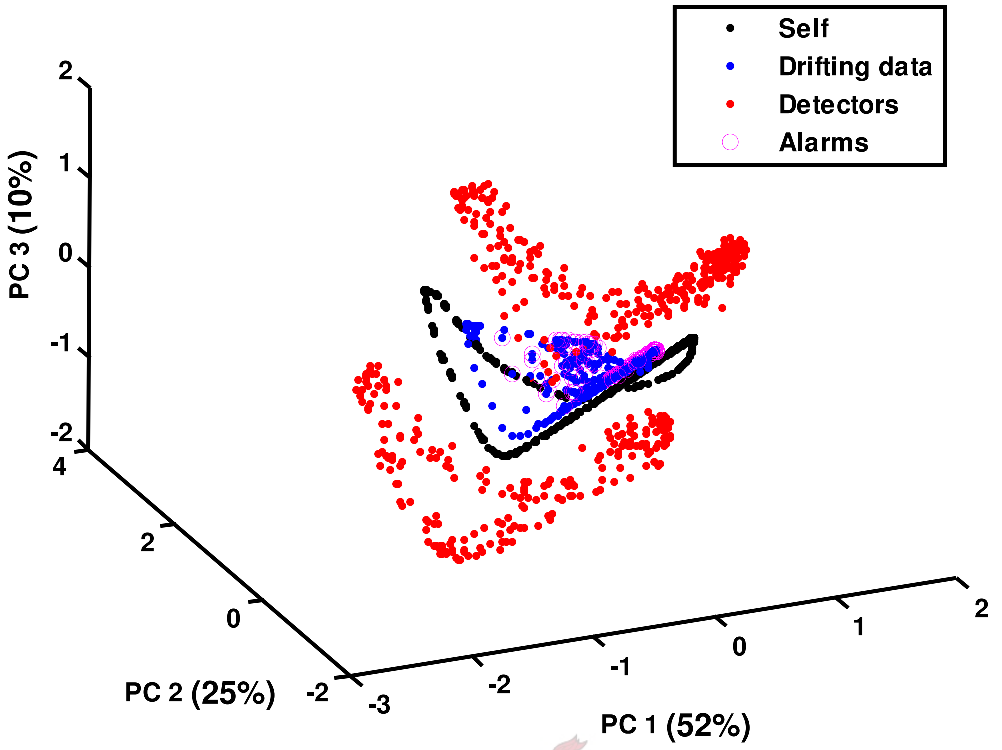


Figure 26: 3-Dimensional presentation of 500 convex hulls detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Once again, ε can be varied to adjust the sensitivity of the detectors. For comparative reasons, ε is once more varied between 0.1 and 0.5 and the results are presented in Table 4.

Table 4: Summary of AIS detection capability for the Predator-Prey data, monitored by 500 convex hulls detectors.

Run	ε	Alarms Triggered		
		Validation (s)	Drifting (m1)	Drifted (m2)
1	0.1	11.9%	42.1%	17.6%
2	0.2	9.0%	55.5%	75.8%
3	0.3	7.9%	81.0%	87.2%
4	*0.4*	*2.2%*	*76.7%*	*92.4%*
5	0.5	4.3%	69.3%	87.3%

Originally, ε was estimated as 0.4 (indicated in *italic*); and from Table 4 it seems as if the estimation is fair. A ROC graph however, clarifies this matter and is therefore shown in Figure 27.

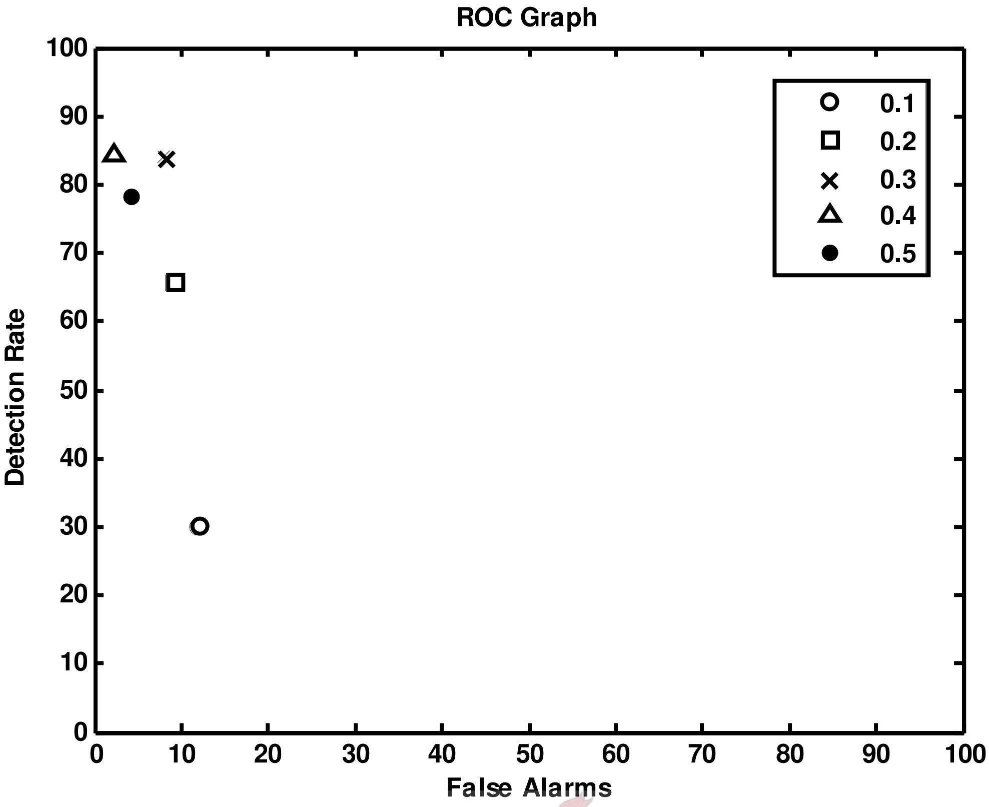


Figure 27: The ROC graph, comparing various different values for ε during random generation.

4.3.3 Hypercube Vertex

The Hypercube Vertex Approach for generating detectors is an approach developed in this project to enhance the efficiency of the already promising concept of AIS. The methodology behind the hypercube vertex method is discussed in Chapter 3, from which it is clear that the number of detectors is variable by varying the interval (n) in which detectors are generated around self events. The order in which the two model parameters (number of detectors, N; and detector affinity threshold ε) are determined is not important in this method since they are not related. To stress this fact, they will intentionally be determined inversely to the other two methods, i.e. the number of detectors are determined first. Figure 28 shows the number of detectors generated, given the generation interval. Considering that 500 detectors were generated in the previous two methods, a choice now exists between n=7 and n=11. For n=7, 645 detectors are generated, while for n=11, 413 detectors are generated. The choice of n=11 is appropriate for two reasons: it minimises the difference in the number of detectors and it will reduce computational strain. Choosing n=11 will not maximise the detection of anomalies though, however, it will test the ability of the hypercube vertex method to detect anomalies with less detectors. Using fewer detectors provides a “worst case scenario” answer to the question of detection capability.

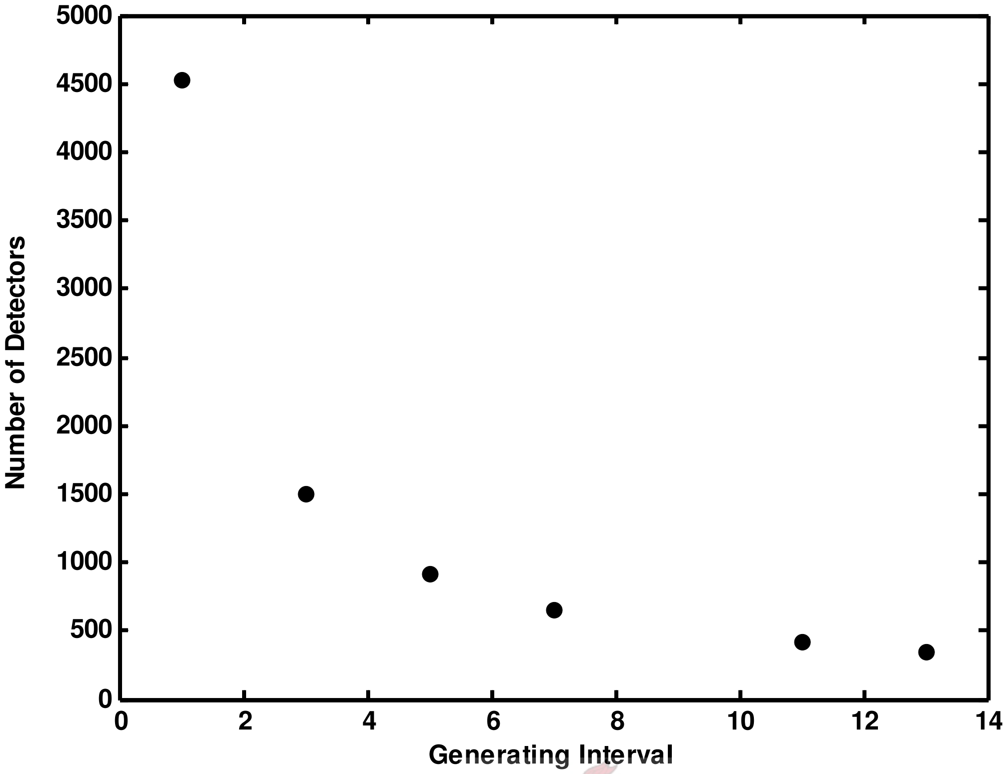


Figure 28: Number of detectors generated using the Hypercube Vertex method versus generating interval.

Experiments have shown that ε can be determined in the same way as in the previous two methods. One difference though, is that ε should be increased by a constant factor of 1.67. This factor was determined empirically by testing various different values of ε, shown in Table 5.

The shape space can now be constructed showing self, nonself, detectors and immune responses (refer to Figure 29).

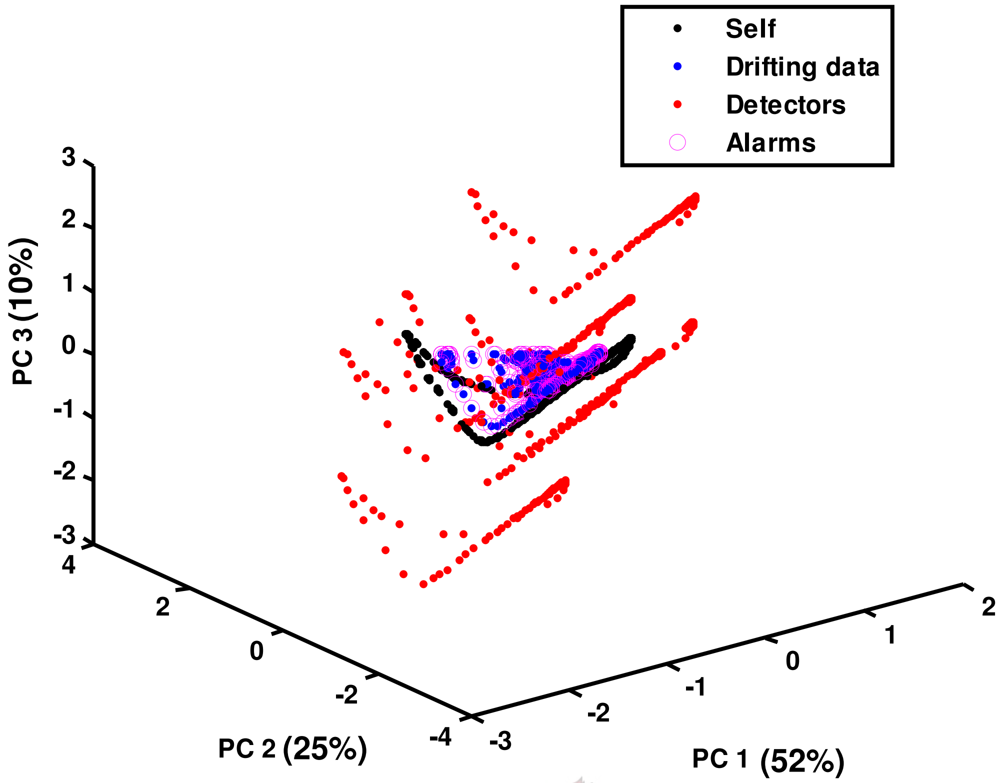


Figure 29: 3-Dimensional presentation of 413 Hypercube Vertex detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Detectors that are generated using the Hypercube Vertex method is placed in strategic positions around the self space to enable them in identifying any process anomaly that drifts from the self space. They form an impassable layer which is located far enough from the self space to allow for noise and minor irregularities. Figure 29 shows the placement of these detectors in 3 dimensions. These specific placements eliminate the need to cover the entire nonself space with detectors. As found in the previous methods of detector generation, ε can be varied to adjust performance. In this case, it was found that ε should be 67% larger than the other methods (i.e. 0.4 x 1.67 = 0.6). The reason being: detectors are placed in layers and ε (radius of coverage) needs to be large enough to cover the areas between layers. Table 5 holds experimental results which prove the need for a larger ε.

Table 5: Summary of AIS detection capability for the Predator-Prey data, monitored by 413 Hypercube Vertex detectors.

Run	ε	Alarms Triggered		
		Validation (s)	Drifting (m1)	Drifted (m2)
1	0.1	2.2%	81.5%	79.5%
2	0.2	1.1%	88.1%	88.8%
3	0.3	0.7%	95.4%	100%
4	0.4	0.0%	98.1%	100%
5	0.5	0.4%	100%	100%
6	*0.6*	*0.0%*	*100%*	*100%*

In the previous two methods, the optimum configuration for ε is 0.4, whereas for the Hypercube Vertex method, the optimum configuration (indicated in *italic*) is ε = 0.6. Auxiliary experiments indicated that the optimum configuration for ε is 67% larger than the calculated ε as per other methods. Figure 30 contains the ROC graph on the various different

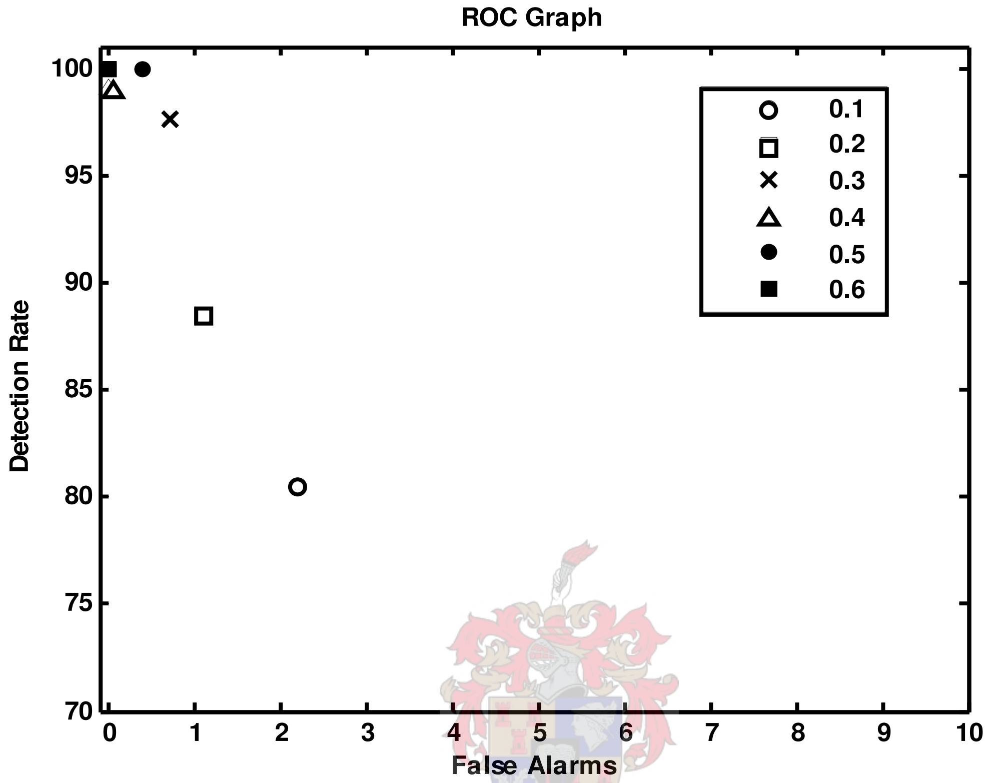


Figure 30: The ROC graph, comparing various different values for ε during the Hypercube Vertex Approach.

The axes on the ROC graph (Figure 30) were adjusted to better utilize the area of the graph. In the case ε=0.6, a perfect classifier is observed, with 100% detection rate and 0% false alarm rate. The Hypercube vertex method is clearly a more advanced method than the previously discussed methods. This statement will undergo intensive examination in subsequent chapters, ultimately leading to credible conclusions. For further reading, refer to Yang et al. (2006).

4.4 2-Dimensional Shape Space

In order to visualise the detector coverage, one can view the shape space in 2 dimensions. The detector affinity threshold ε is visualised by plotting circles of radius ε around each detector. Some of the variation in the data is sacrificed by considering only the first two principal components. Figure 31 shows the 2-dimensional shape space of the predator-prey data. Three figures illustrate different methods for generating detectors and their coverage of the nonself space.

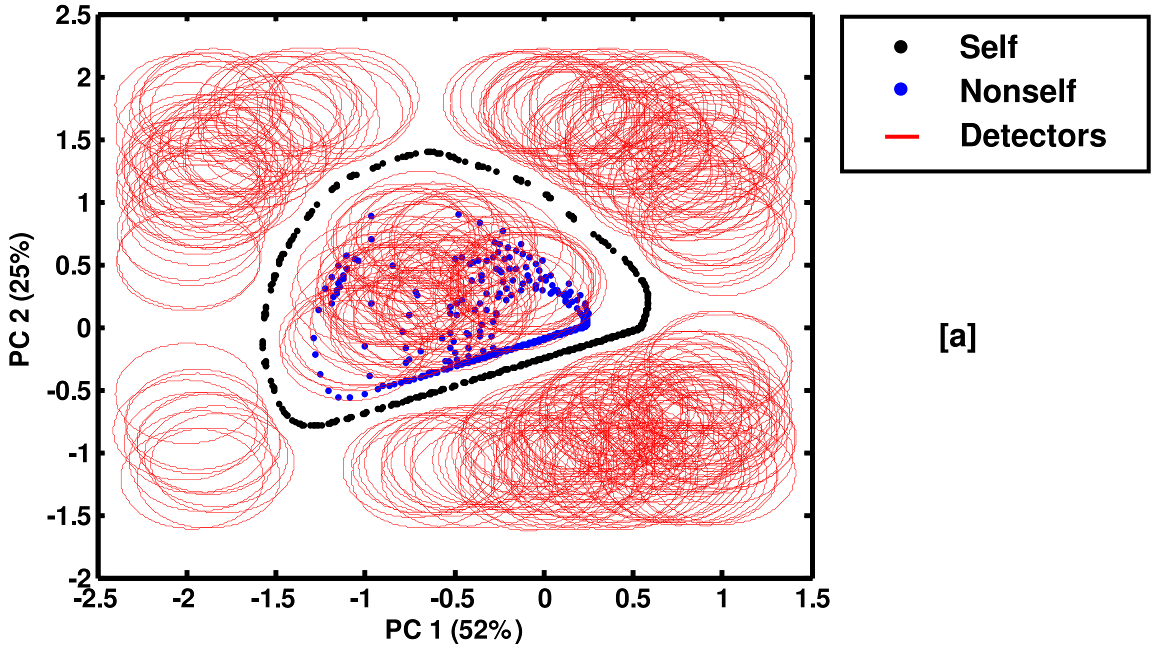


Figure 31 [a]: 2-Dimensional shape space of the predator-prey data, visually showing the detector affinities for 500 random detectors with ε = 0.4.

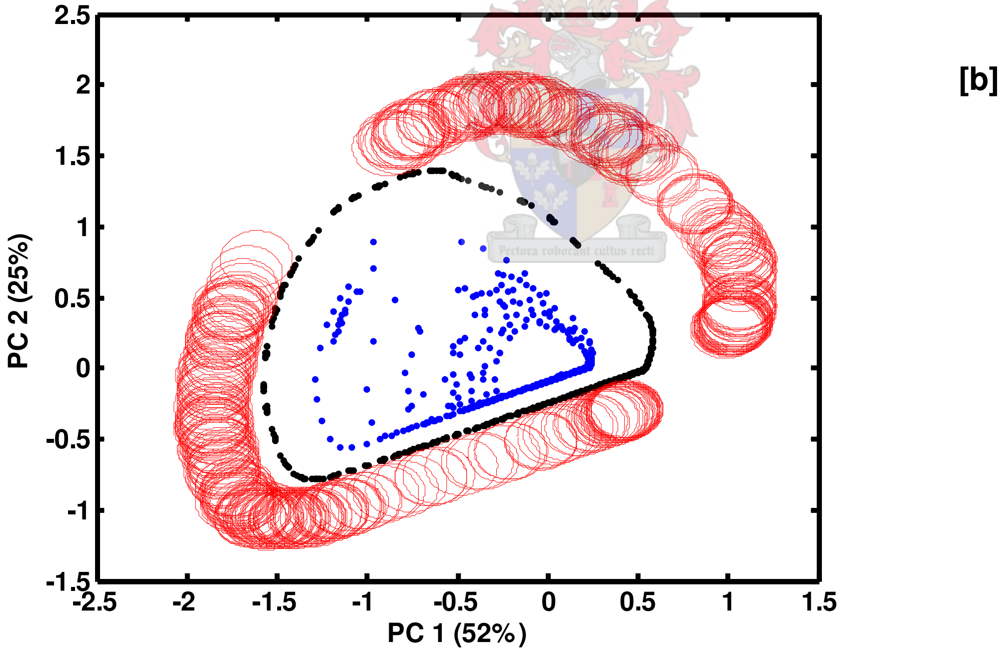


Figure 31 [b]: 2-Dimensional shape space of the predator-prey data, visually showing the detector affinities for 500 convex hulls detectors with ε = 0.4.

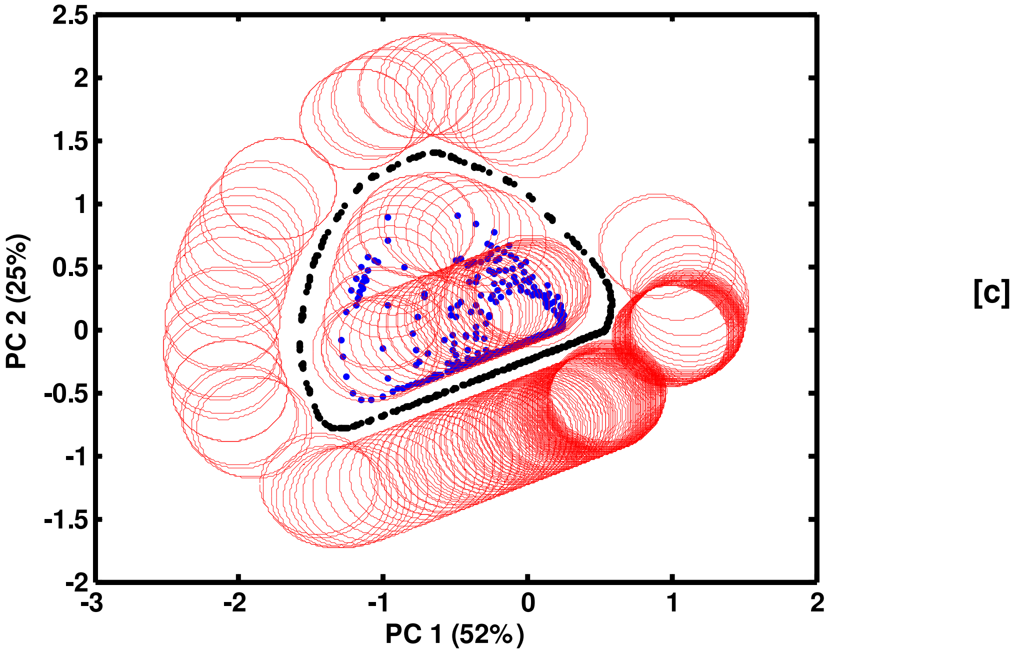


Figure 31 [c]: 2-Dimensional shape space of the predator-prey data, visually showing the detector affinities for 500 hypercube vertex detectors with ε = 0.6.

For each method, the optimum detector affinity threshold and number of detectors, as discussed earlier, is chosen. For random generation (Figure 31 [a]) a section of the nonself space above and to the left of the self space is not covered. Aside from this, the largest part of the nonself space is covered. For the hypercube vertex method on the other hand (Figure 31 [c]) this section, above the self space is covered, but a section to the right on the self space is left uncovered. The important section of the nonself space (inside the self space) is however, well covered. There are two basic reasons for sections of the nonself space not being covered, they are:

a) No detector was placed in this area.

b) A detector was placed here but was removed in the screening process due to a self event falling in its affinity threshold.

The gaps in the covered nonself areas shown in Figure 31 can now be explained using these two causes. In random generation (Figure 31 [a]) it is possible but unlikely that no detectors were placed in these gaps in the first place. A more likely explanation is that the detectors placed in these gaps were located too close to the self space and were therefore censored. This argument is supported by the fact that the gaps are all located where the self space falls closest to the shape space boundaries, i.e. the northern, eastern, southern and western points of the self space. In generation using convex hulls (Figure 31 [b]) the cause for the gaps is possibly a combination of the two reasons stated above. On the north-western side of the self space, it is clear that the detectors falling in these gaps will coincide with the self space. On the south-eastern side, however, it is more likely that detectors were not placed in this area in the first place due to the nature of the convex hulls drawn. During generation using the hypercube vertex approach (Figure 31 [c]), the likely explanation for the gap on the north-eastern side is that no detectors were placed here. The fact that every n^{th} self event is used to produce detectors may cause a certain area of the self trajectory to be missed. It is important to realise that neighbouring points in the self trajectory does not necessarily follow a chronological order. It is more likely that neighbouring points are placed next to each other on a different passing of the trajectory. It is therefore possible that a certain region may be omitted when not generating detectors on every self event.

4.5 Effects of Noise on the Predator-Prey System

Noise is an unavoidable factor that any fault identification system must take into account. The influence of noise is studied on all three case studies as seen in Chapters 4, 5 and 6. Examining the effect of noise on all generating methods is tedious and nonsensical, since the impact of noise on a given method is proportional to that of any supplementary method. The Hypercube Vertex method was therefore chosen to study the impact of noise. This section investigates the influence of noise on the Hypercube Vertex approach in the Predator-Prey system. Gaussian noise is added at 10% of the standard deviation of the measurement space of the Predator-Prey system. All parameters, i.e. number of detectors and detector affinity threshold ε, are kept constant at their optimum values as shown above.

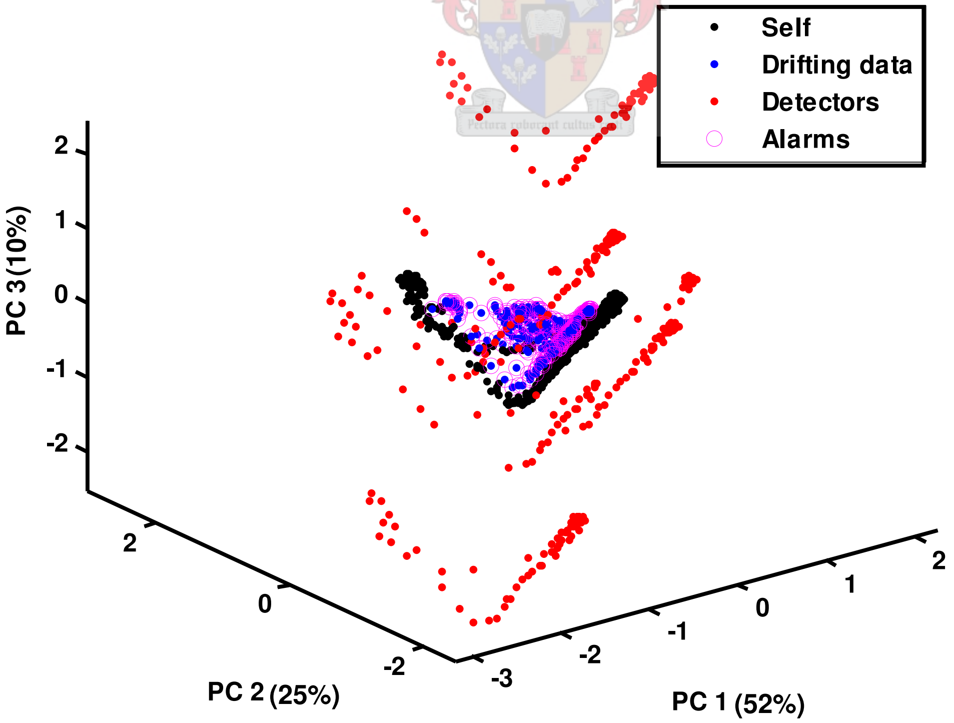


Figure 32: The shape space of the Predator-Prey system with noise added at 10% of the standard deviation.

Figure 32 shows the shape space of the Predator-Prey system with 10% noise. As indicated, the first three principal components capture 87% of the data variation. It is also clear that the self space of the predator prey system is disrupted and no clear boundaries can be drawn. This causes the detectors to scatter from their usual trajectory. Nevertheless, 95.7% of the drifting data is identified as anomalous, while 100% of the drifted data are correctly classified as anomalous. This gives an average detection rate of 97.9%. A mere 0.7% false alarm rate is observed. These figures were obtained by retaining the previously determined values for all parameters, i.e. 500 detectors with $\varepsilon = 0.6$. The perfect classifier is not witnessed in the case where noise disturbs the signal, but the detection is exceptional nonetheless. A summary of the classification capability of the various methods of generating detectors on the different case studies is given in Figure 65 by use of a ROC graph.

4.6 Summary of Chapter 4

This chapter discusses the nature of a Volterra process and focuses on the predator-prey system. The model for the predator-prey system is presented; simultaneously identifying the drifting model parameters are identified. Figure 18 shows a representation of the two steady states under consideration in the predator-prey equilibrium. The transition between these two states is not shown, due to the size of the dataset (5000 time units).

The transformation between measurement-space and shape-space is discussed, considering the autocorrelation function. The autocorrelation lag is established at 6 data increments, which determines the dimensionality of the system. PCA is applied and according to the latency values, 87% of the trend can be captured by viewing only 3 dimensions. After constructing the self-space, detectors are generated using various methods. By estimating the detector coverage, it was decided that 500 detectors adequately cover the nonself space for a detector affinity threshold of 0.4. The various methods are discussed and their performances are shown in associating ROC graphs. A combined ROC graph is shown in Figure 33, which visibly illustrates the performance of each method for generating detectors. A perfect classifier is one that lies at the coordinates (0,100) on the ROC graph, as is found for one of the Hypercube Vertex runs. Furthermore it is clear that the average performance of the Hypercube Vertex method is a significant improvement on both Convex Hulls and Random Generation, which substantiates the superiority of the method.

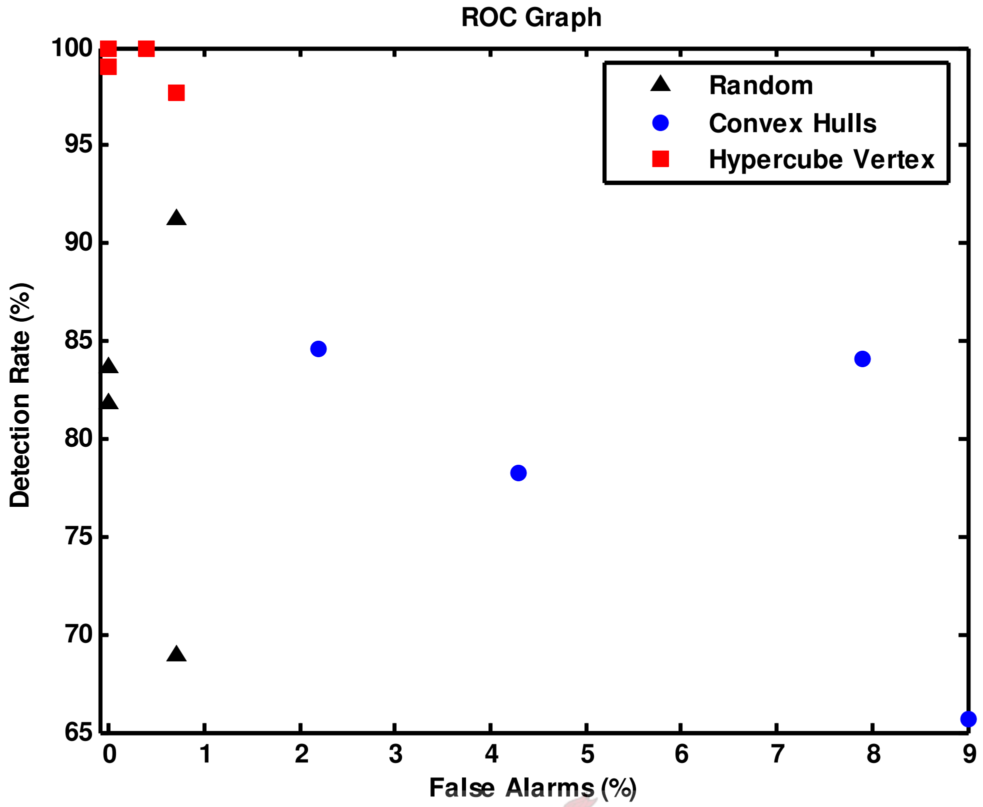


Figure 33: A combined ROC graph comparing the performance of three different methods to generate detectors for the predator-prey equilibrium.

Chapter 5: Applying AIS: An Autocatalytic Reaction

5.1 Introduction

The second case study under consideration is a system of two parallel, isothermal autocatalytic reactions taking place in a continuous stirred tank reactor (CSTR) (Lee et al., 1996). This process is commonly used in testing diagnostic classifiers. The kinetics of the system proceeds according to the following steps:

$$\begin{aligned} &A + 2B \rightarrow 3B \\ &B \rightarrow C \\ &D + 2B \rightarrow 3B \end{aligned} \qquad [17\text{ a-c}]$$

The reactions above are governed by the following rate equations:

$$\begin{aligned} -r_A &= k_1 C_A C_B^2 \\ -r_C &= k_2 C_B \\ -r_D &= k_3 C_D C_B^2 \end{aligned} \qquad [18\text{ a-c}]$$

When the reaction is carried out in a CSTR, the system can be described by three ordinary differential equations, given below:

$$\begin{aligned} \frac{dx_1}{d\tau} &= 1 - x_1 - D_{a1} \cdot x_1 \cdot x_3^2 \\ \frac{dx_2}{d\tau} &= 1 - x_2 - D_{a2} \cdot x_2 \cdot x_3^2 \\ \frac{dx_3}{d\tau} &= 1 - (1 + D_{a3})x_3 + \gamma_1 D_{a1} \cdot x_1 \cdot x_3^2 + \gamma_2 D_{a2} \cdot x_2 \cdot x_3 \end{aligned} \qquad [19\text{ a-c}]$$

The dimensionless concentrations are defined by:

$$x_1 = \frac{C_A}{C_{A0}} \qquad x_2 = \frac{C_D}{C_{D0}} \qquad x_3 = \frac{C_B}{C_{B0}} \qquad [20\text{ a-c}]$$

The Damköhler numbers for species A, D and B, ratios of the species in the feed and dimensionless time are defined by:

$$D_{a1} = \frac{k_1 C_{B0}^2 V}{Q} \qquad D_{a2} = \frac{k_3 C_{B0}^2 V}{Q} \qquad D_{a3} = \frac{k_2 V}{Q}$$

$$\gamma_1 = \frac{C_{A0}}{C_{B0}} \qquad \gamma_2 = \frac{C_{D0}}{C_{B0}} \qquad \tau = \frac{t \cdot Q}{V} \qquad [21\text{ a-f}]$$

The model parameters were chosen as:

$$D_{a1} = 18000 \qquad D_{a2} = 400 \qquad D_{a3} = 80$$

$$\gamma_1 = 3.5 \qquad \gamma_2 = 6.2$$

A representation of the reaction equilibrium is given in Figure 34, which is obtained by means of the Runge-Kutta method. The anomalous data are obtained by allowing γ_1 and γ_2 to drift in 10 increments of 0.02. Each increment is allowed time to reach steady state before the next is initialised. The final values for γ_1 and γ_2 would therefore be 3.7 and 6.4 respectively. Three datasets are generated, each containing 5000 data points separated by a sampling rate of 0.05.

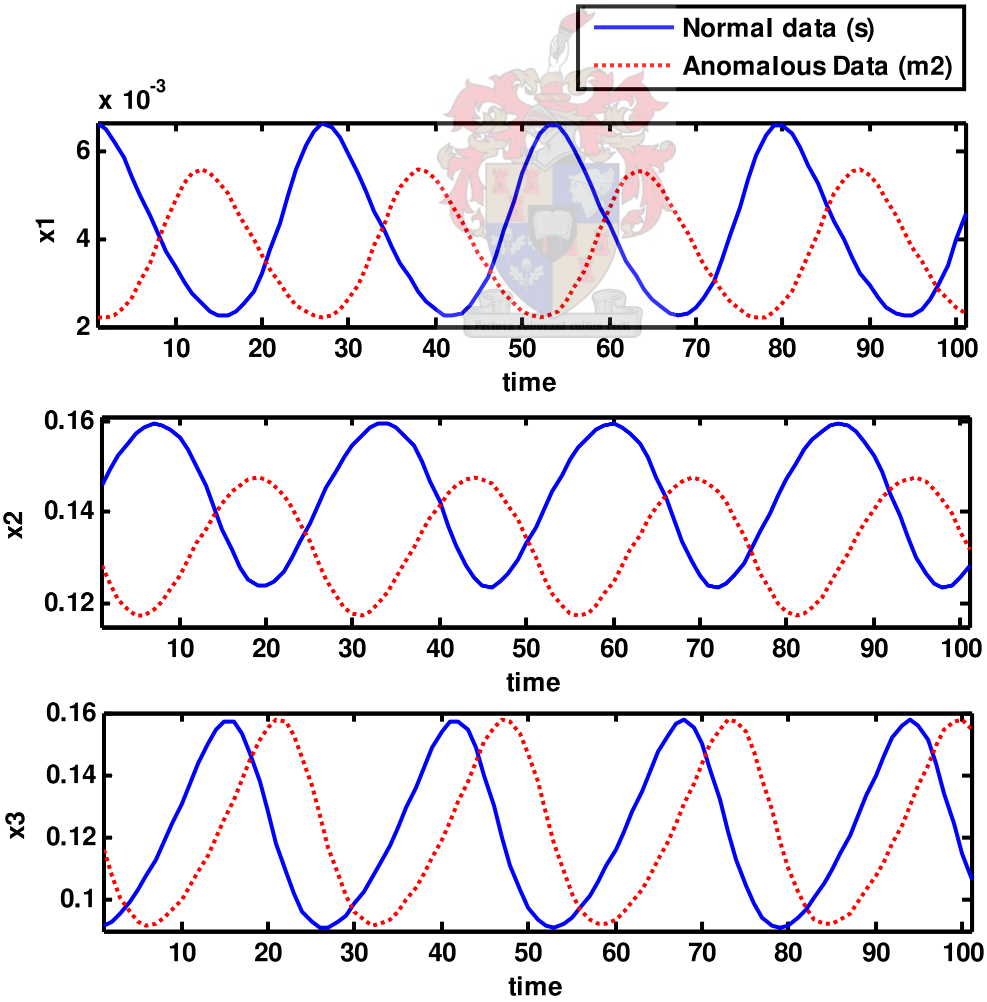


Figure 34: Representation of the autocatalytic reaction equilibrium: The first steady state is indicated in a solid line and represents normal process behaviour; the second steady state is

shown in a dotted line and represents anomalous behaviour.

5.2 Transforming between measurement space and shape space

Transforming the time series data, s (measurement space) into the shape space S, requires the use of the autocorrelation lag, see Figure 35.

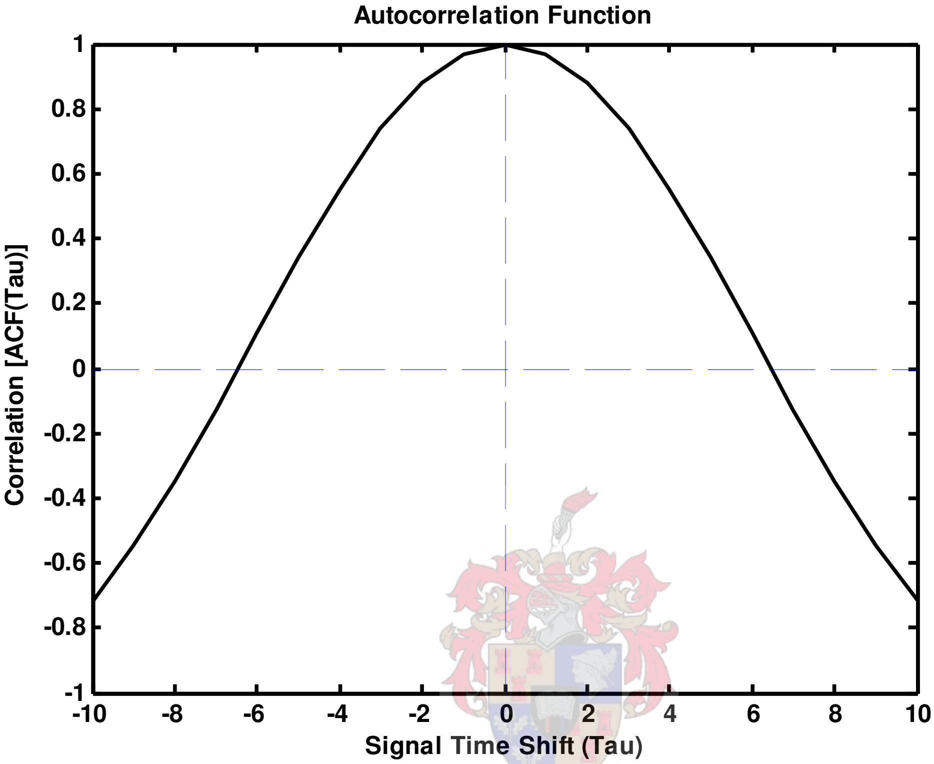


Figure 35: Autocorrelation function of time series data s for the autocatalytic reaction.

Once again, only the first time series needs to be analysed, after which all the times series' lags are considered equal. From Figure 35 a signal time shift (or lag) of 7 data points provides an uncorrelated transformation. The data are mean centred and scaled prior to performing the principal components analysis. Figure 36 holds the latencies of the principal components of the self-space. The first 2 dimensions accurately describe 98.9% of the data. In Chapter 2 it is mentioned that an accuracy of around 90% is sufficient in representing the data. It is therefore decided that 2 dimensions is quite adequate for this case study.

Figure 37 shows the 2-dimensional shape-space of the autocatalytic reaction. It differs from Predator-Prey system in that the points in the trajectory follow a continuous path. This may be due to a relatively smaller sampling rate, but the fact remains that there are no omitted sections that give rise to holes in the trajectory. It is consequently expected that ε will be somewhat smaller in this case as opposed to the Predator-Prey system.

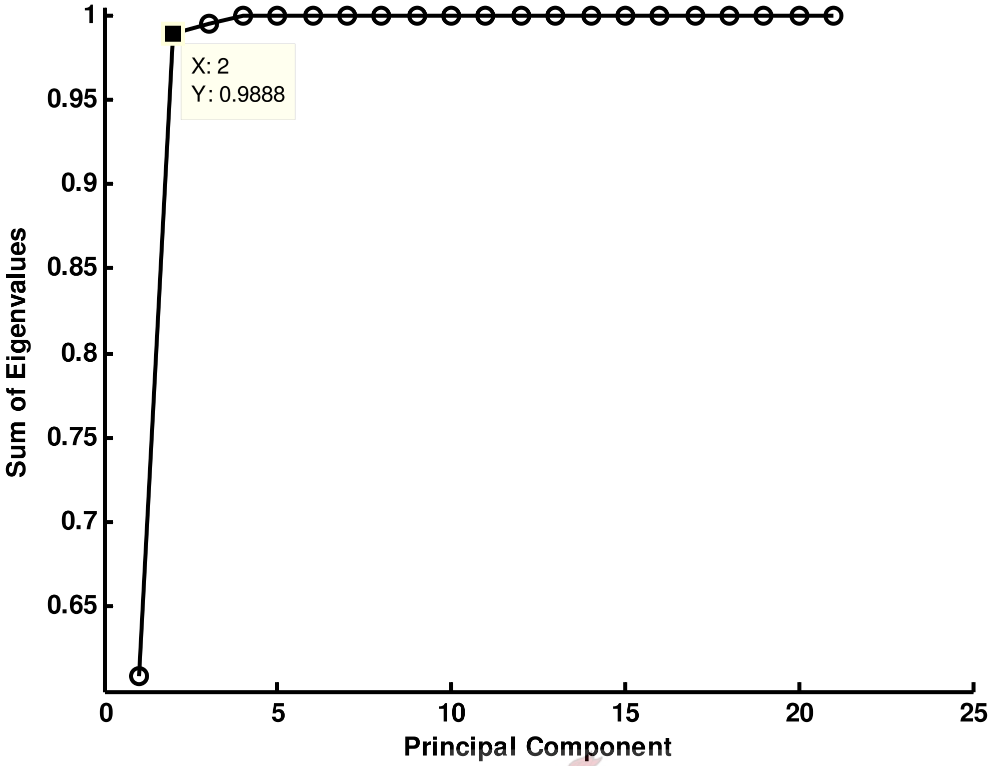


Figure 36: Cumulative sum of latencies of the principal components of self space S of the autocatalytic reaction data.

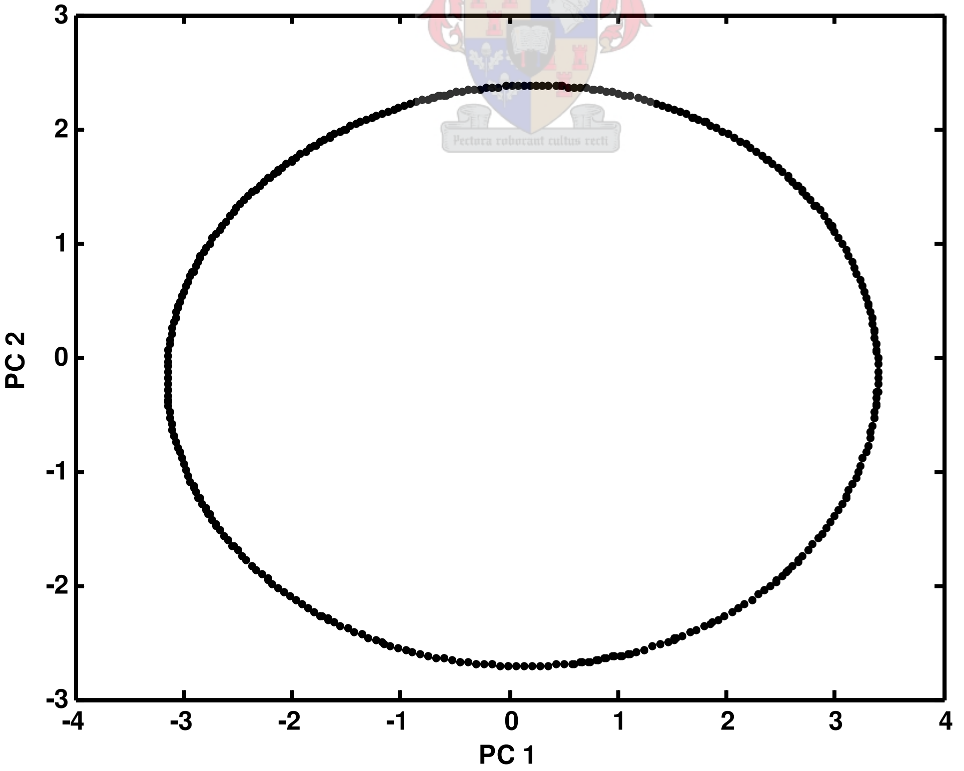


Figure 37: 2-Dimensional Principal Components plot of the Self-Space of the autocatalytic reaction data

5.3 Generating Detectors and Classifying Data

Since the self space is now known, the generation of detectors may commence. Once again, the three methods for generating detectors are investigated and their performances are compared. The detector affinity threshold ε is again varied and its influence is illustrated by monitoring the nonself space. It is important to realise that varying ε does not form part of training AIS, since no knowledge of the nonself space is available in this stage.

5.3.1 Randomly Generated Detectors

To generate detectors randomly, as is the case in the previous chapter, there are two variables that need be determined. Once again, they are the affinity threshold, ε and number of detectors, N. The magnitude of ε is determined, as discussed in Chapter 3, by considering the distance between neighbouring events in the self space. This magnitude is varied to confirm its validity and results are shown in Table 6. Due to the smaller sampling rate of the autocatalytic data (0.05 as opposed to the sampling rate of 0.1 in the predator-prey data), the scaling factor between the distance between neighbouring events and ε is adjusted to a full order of magnitude.

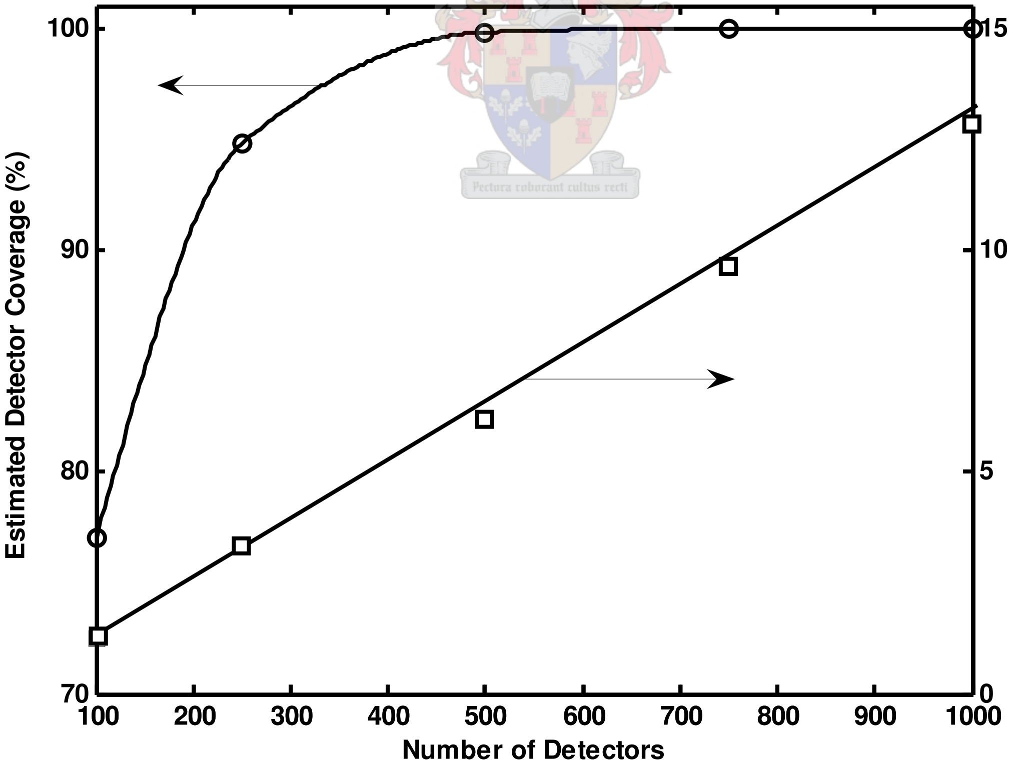


Figure 38: Estimated random detector coverage and time taken to generate these detectors as a function of the number of detectors for ε = 0.4.

To solve the number of detectors, the coverage of the nonself space is estimated as a function of the number of detectors. Generating a set of detectors involves a certain computational requirement, which is measured by the time needed to generate this set of detectors. Figure 38 shows two trends that illustrate the estimated detector coverage and the time needed to generate a number of detectors. From Figure 38 it seems as if 400 detectors is a fair approximation towards an optimisation between computational strain and estimated detector coverage. It is estimated that around 98% of the nonself space is covered by 400 detectors, while generating these detectors takes approximately 5 seconds.

Figure 39 shows the shape space, complete with self, nonself detectors and identified process anomalies. Note that Figure 39 is constructed with ε = 0.4 for illustrative purposes. The value for ε is varied and results summarised in Table 6.

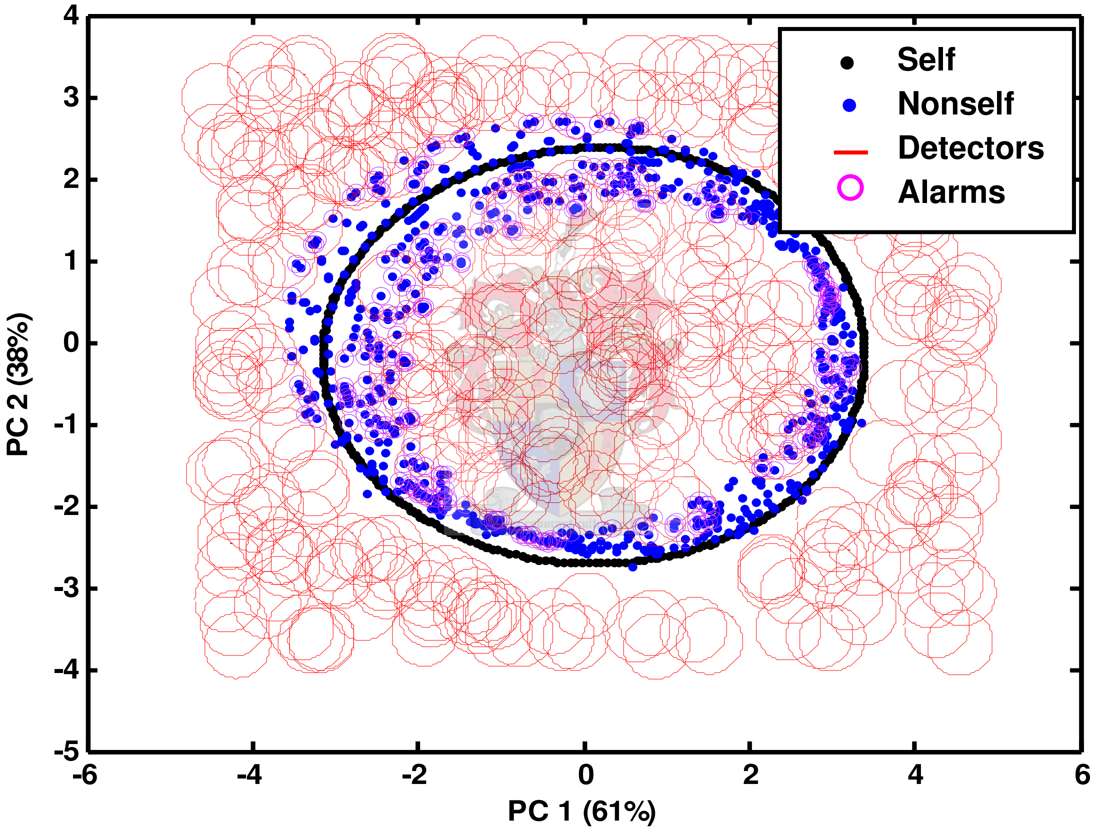


Figure 39: 2-Dimensional presentation of 400 random detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Table 6 is an example of the validation process for the parameter ε. It is recommended that this process is done for each case of randomly generated detectors.

Table 6: Summary of AIS detection capability for the autocatalytic reaction data, monitored by 400 randomly generated detectors.

Run	ε	Number of detectors	Alarms Triggered		
			Validation (s)	Drifting (m1)	Drifted (m2)
1	0.2	2000	0.4%	34.2%	58.7%
2	0.3	900	0.0%	36.5%	43.8%
3	0.4	400	0.6%	37.3%	59.2%
4	0.5	*300*	*0.0%*	*33.9%*	*60.9%*
5	0.6	250	0.0%	31.1%	50.6%

ε = 0.5 appears to be the optimum (shown in *italic*) configuration under random generation when considering both Table 6 and Figure 40.

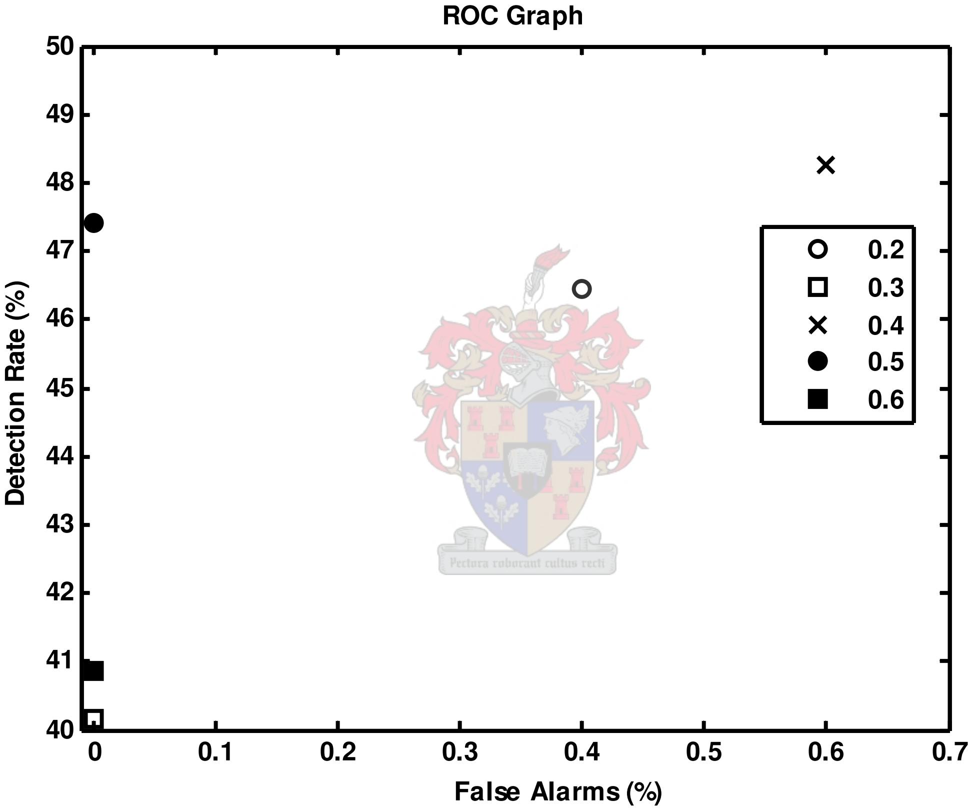


Figure 40: The ROC graph, comparing various different values for ε during random generation.

5.3.2 Convex Hulls

When using the theory of convex hulls to generate detectors, the detector coverage once again plays a role in determining the optimum number of detectors. As discussed in Chapter 4, the meaning of the estimated coverage is not exactly related to its meaning when considering random detectors. Since the detectors are clustered together in a smaller space, 100% estimated coverage will be reached with far less detectors. It does, however, give an approximation as to how many uncovered regions exist in the detector field. Figure 41 shows the estimated detector coverage and time needed to generate a given number of detectors.

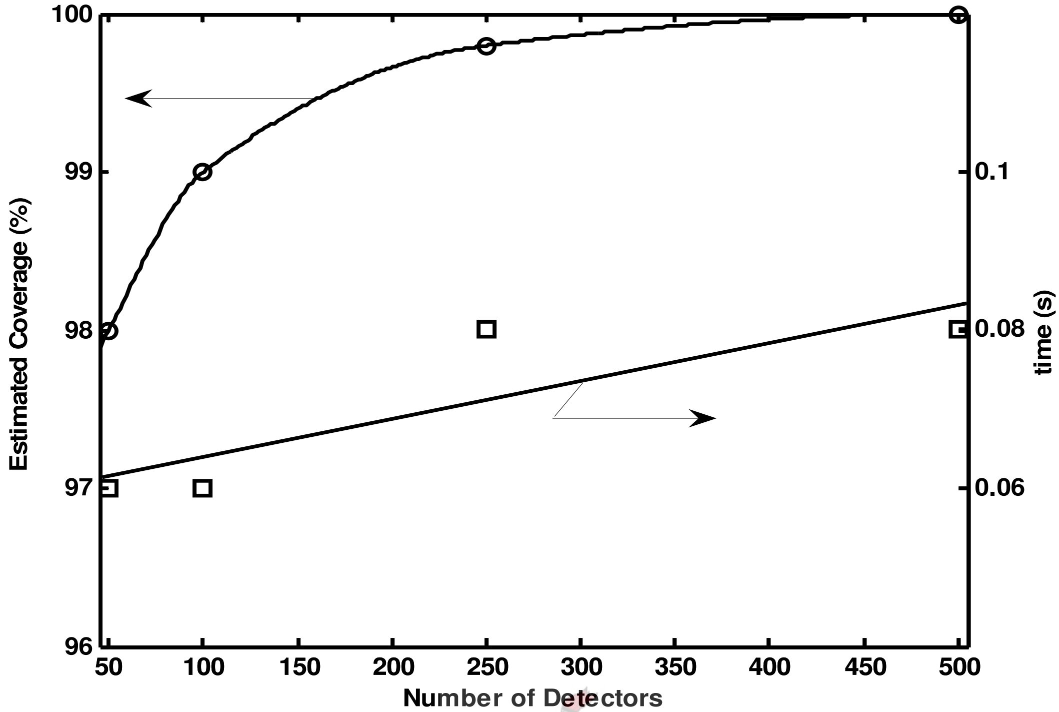


Figure 41: Estimated convex hull detector coverage and time taken to generate these detectors as a function of the number of detectors for ε = 0.4.

From Figure 41 it is apparent that, as is the case with the Predator-Prey data, the time required to generate a given number of detectors is insignificant. It is therefore a logical deduction that the maximum number of required detectors may be chosen without exceeding practicable computational strain. Once again, as compared to random detectors, 400 detectors seem adequate with an estimated coverage in this case of approximately 99.8%. Bearing this in mind, the detectors are generated and shown in Figure 42 along with the self space, nonself space and identified process anomalies.

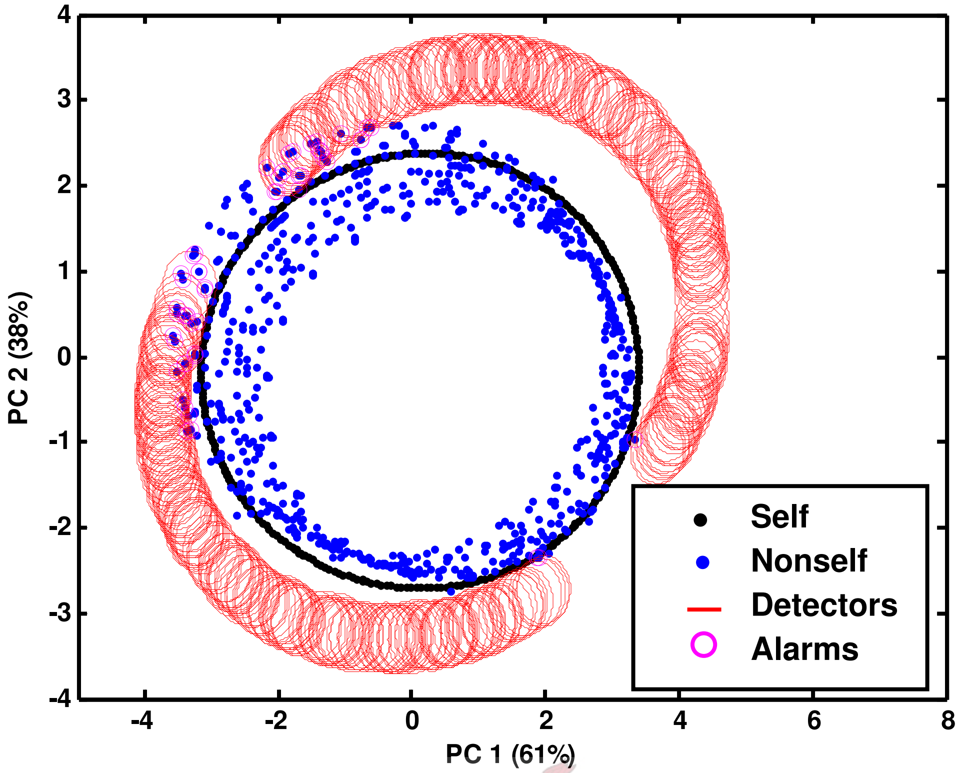


Figure 42: 2-Dimensional presentation of 400 convex hulls detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

On inspection of Figure 42 it seems as if the convex hulls detectors' fault identification capability is lacking. This can be established by altering the calculated value of ε, as shown in Table 7. Once again, there are gaps in the trajectory of the detectors. In this case, it is clear that these gaps are formed due to post generation screening of detectors. Connecting these gaps with detectors will cause an overlap of the self events and the detector affinity thresholds.

Table 7: Summary of AIS detection capability for the autocatalytic reaction data, monitored by 400 convex hulls detectors.

Run	ε	Alarms Triggered		
		Validation (s)	Drifting (m1)	Drifted (m2)
1	0.1	0.0%	2.5%	0.0%
2	0.2	0.0%	2.1%	0.0%
3	0.3	0.0%	1.3%	0.0%
4	0.4	0.0%	0.6%	0.0%
5	*0.5*	*0.0%*	*0.8%*	*0.0%*

Unlike with the predator-prey data, the method of convex hulls does not appear to identify any process anomalies. Figure 43 shows the ROC graph for the different values of ε as in Table 7. The detection rate is negligible and this method is therefore not recommended for monitoring the autocatalytic reaction.

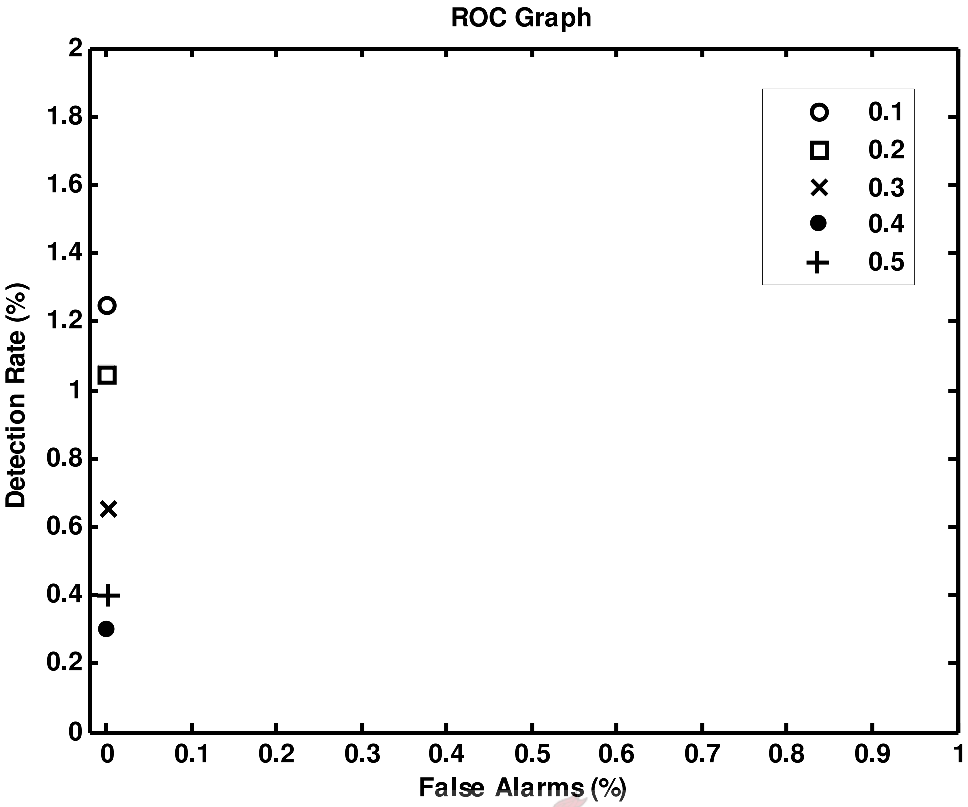


Figure 43: The ROC graph, comparing various different values for ε during random generation.

Thus far, no significant success has been achieved in monitoring the autocatalytic reaction. In Chapter 4, the hypercube vertex method is the method of choice, showing outstanding capability for fault detection. The following section assesses the performance of the hypercube vertex method, attempting to improve classification in the autocatalytic reaction.

5.3.3 Hypercube Vertex

With the Hypercube Vertex Method, the number of detectors is not a function of detector coverage. Instead, they are generated in a ratio relative to the size of the self set. Their numbers are controlled by choosing every n^{th} point in the self space. For comparability, the number of hypercube vertex detectors will be limited to an interval around the size of the detector sets of the previous two methods, i.e. 400 ± 50. Figure 44 shows the number of detectors generated as a function of n (generating interval). For n=11, 370 detectors are generated, which is less than 400 but still falls within the tolerable interval. Using less than 400 detectors puts the hypercube vertex method at a disadvantage to the other methods, which in turn will provide a “worst case” answer to the question of detection capability.

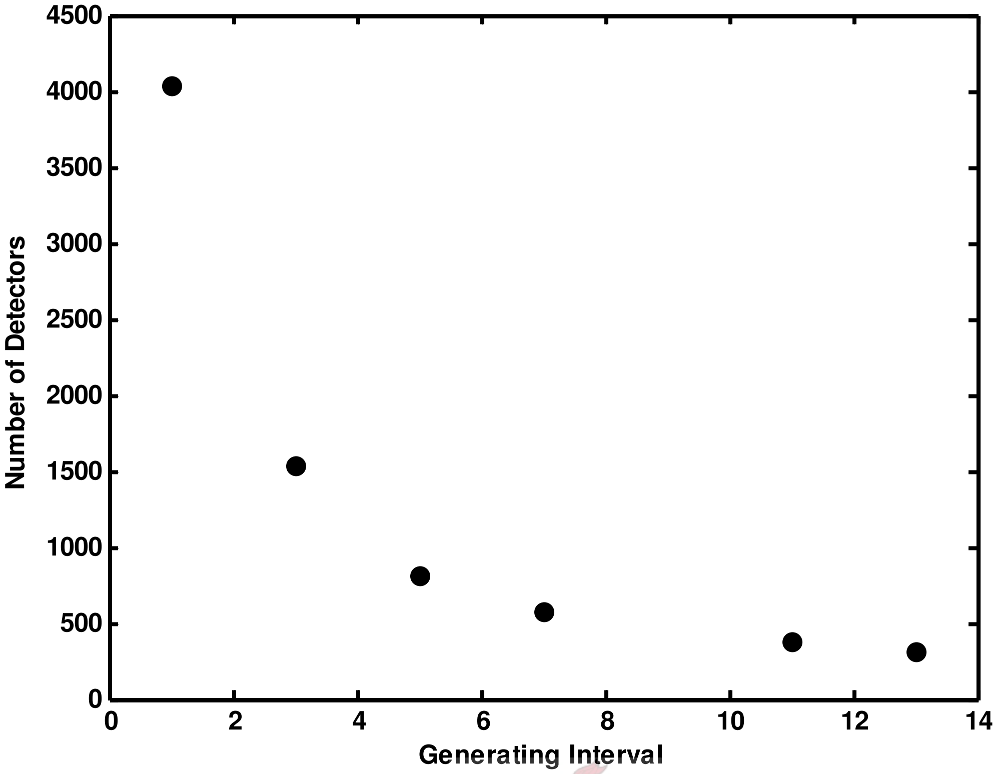


Figure 44: Number of detectors generated using the Hypercube Vertex method versus generating interval.

The detector affinity threshold ε is determined in the same manner as in the previous sections and will once again be varied from 0.1 through 0.5 for validation purposes. Refer to Table 8 for a summary of the performance of the hypercube vertex method. Figure 45 is an illustration of the shape-space of the autocatalytic reaction data, where circles indicate anomalous events.

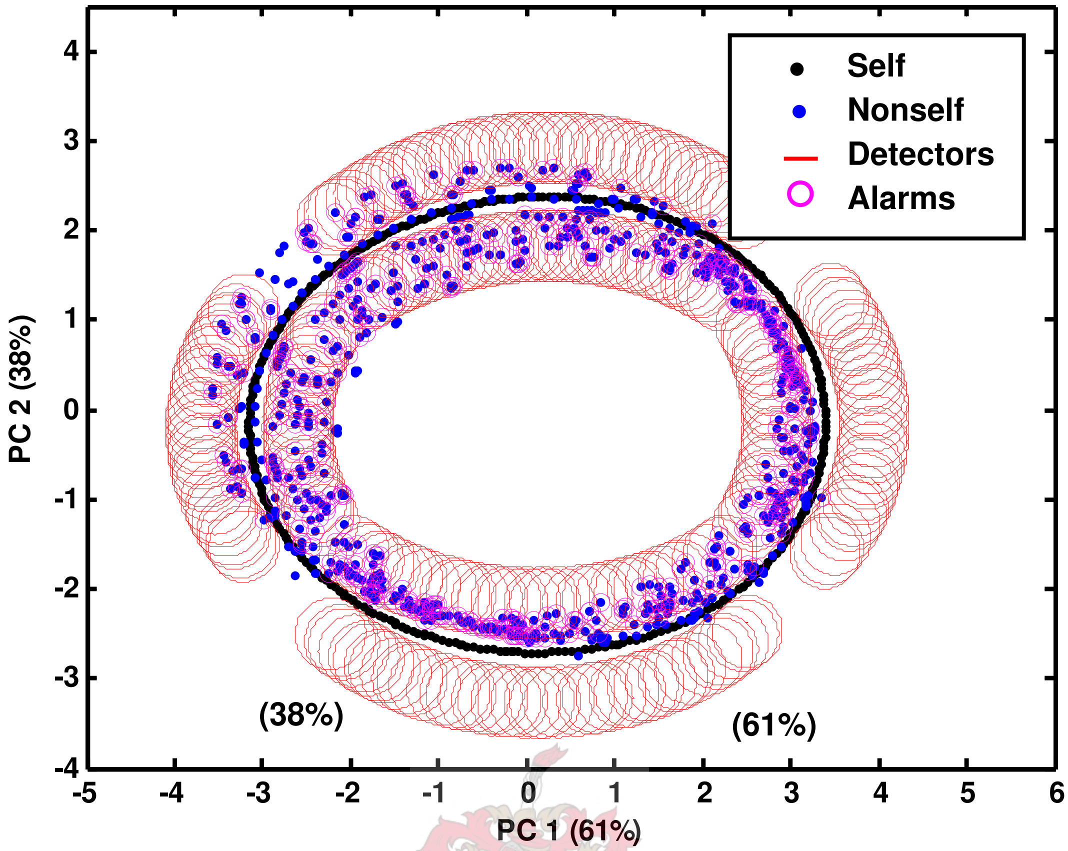


Figure 45: 3-Dimensional presentation of 370 Hypercube Vertex detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

From Figure 45 it appears as if far more anomalous events have been identified compared to the previous two methods. This cannot be conclusively stated prior to formal experimentation, tabulated in Table 8. A more scrupulous conclusion can be made when considering a ROC Graph. From Figure 45 it is clear that the gaps in the detector coverage are not formed by detectors not being placed in these regions. This can be said because the inner detector trajectory is covered. Therefore, the only possible reason for gaps is post generation removal due to detectors falling too close to the self trajectory.

Table 8: Summary of AIS detection capability for the autocatalytic reaction data, monitored by 370 Hypercube Vertex detectors.

Run	ε	Alarms Triggered		
		Validation (s)	Drifting (m1)	Drifted (m2)
1	0.1	1.7%	33.9%	0.0%
2	0.2	1.3%	77.3%	88.9%
3	0.3	0.0%	86.6%	99.6%
4	*0.4*	*1.7%*	*79.8%*	*100%*
5	0.5	0.0%	76.6%	100%

Table 8 shows a remarkable improvement in classification to the previous two methods. The row in *italic* print indicates the chosen optimum parameter setup. The ROC graph is shown in Figure 46 where the optimum value for ε is determined as 0.3.

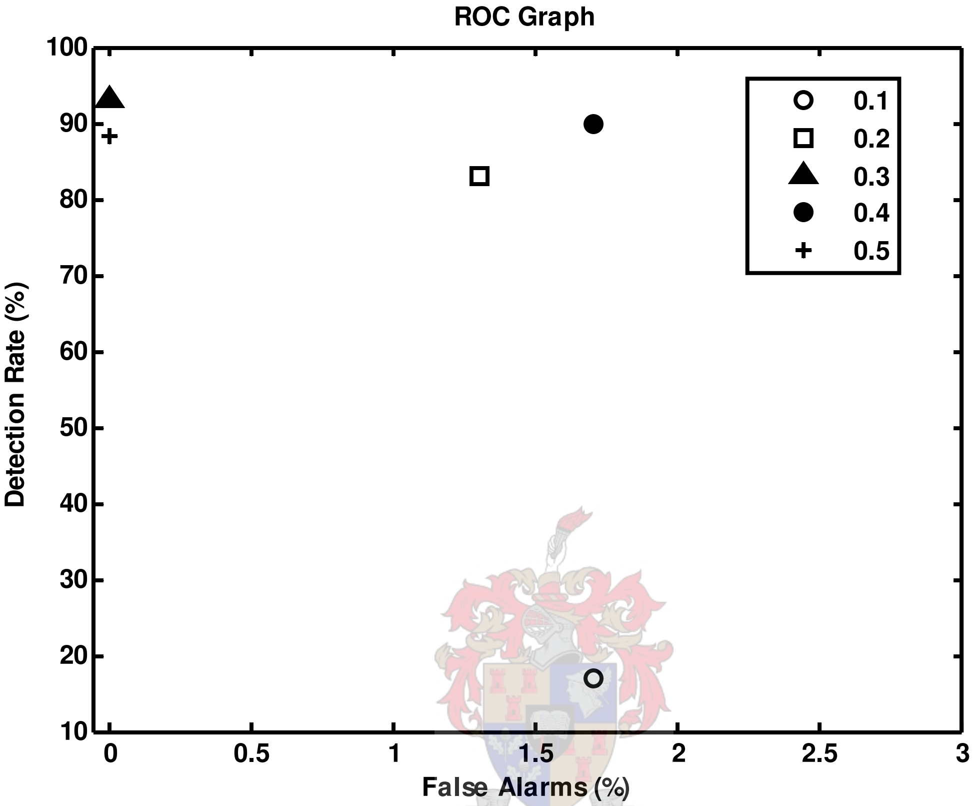


Figure 46: The ROC graph, comparing various different values for ε during the Hypercube Vertex Approach.

The axes of the ROC curve were again adjusted to better utilise the plot area. The hypercube vertex approach again proves its superiority: strongly recommending it as the method of choice.

5.4 Effects of Noise on the Autocatalytic Reaction

As in Chapter 4, noise is added to the system and the Hypercube Vertex method is tested for its ability to handle noise. The Hypercube Vertex method is chosen because it displayed the best performance in the previous sections. Figure 47 shows the shape space of the autocatalytic reaction after addition of noise. This noise is once again added at 10% of the standard deviation of the measurement space. Model parameters are kept at their predicted values as discussed above. Figure 47 shows the shape space including noise. It also shows that the first two principal components describe 92% of the original data. It is therefore, not necessary to add a third dimension, even though noise has been added.

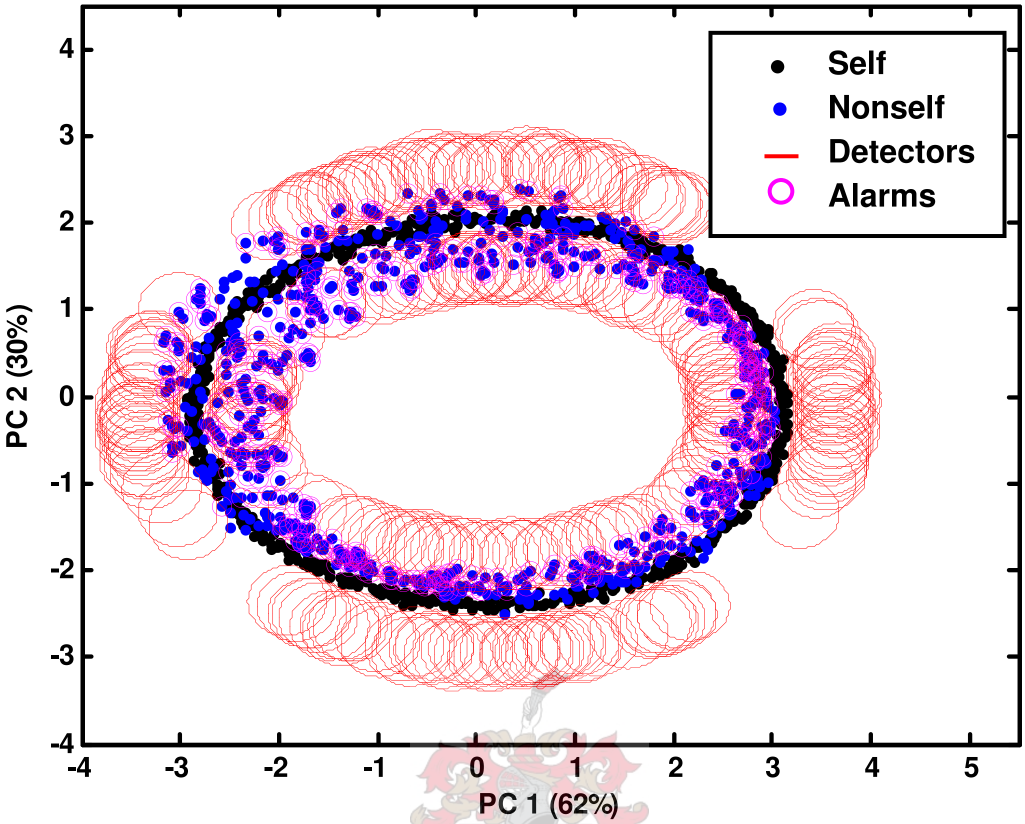


Figure 47: The shape space of the autocatalytic reaction with noise added at 10% of the standard deviation.

The largest impact of noise is observed in the drifting data, where 72% of anomalous events are identified. This may be due to the random scattering of both self and nonself events. Some nonself events are scattered into the self space, which is now larger than what was found without the noise. An irregular boundary is created between the self and nonself space. The second steady state, on the other hand, is situated at a large enough distance from the self space. Noise has less impact on this space and a detection rate of 97.8% is observed. Astonishingly, only 0.4% of the validation set is misclassified, which translates to a low false alarm rate. The average detection rate between sets m1 and m2 is 85.4% which is more than acceptable given 10% noise.

5.5 Summary of Chapter 5

This chapter focuses on an autocatalytic reaction of which the reaction, rate law and defining differential equations are given. It uses the same methods as discussed in the previous chapter, where classification of data into either an anomalous or a normal category was considered. Figure 34 represents the various concentrations in the CSTR at two steady states.

The transformation between the measurement space and shape space is discussed by referring to the autocorrelation function, of which the lag is established as 7 measurements. After PCA is applied, 98.9% of the data variation is retained and 2 dimensions are accepted. The various methods of generating detectors are discussed and their performances are measured in ROC graphs.

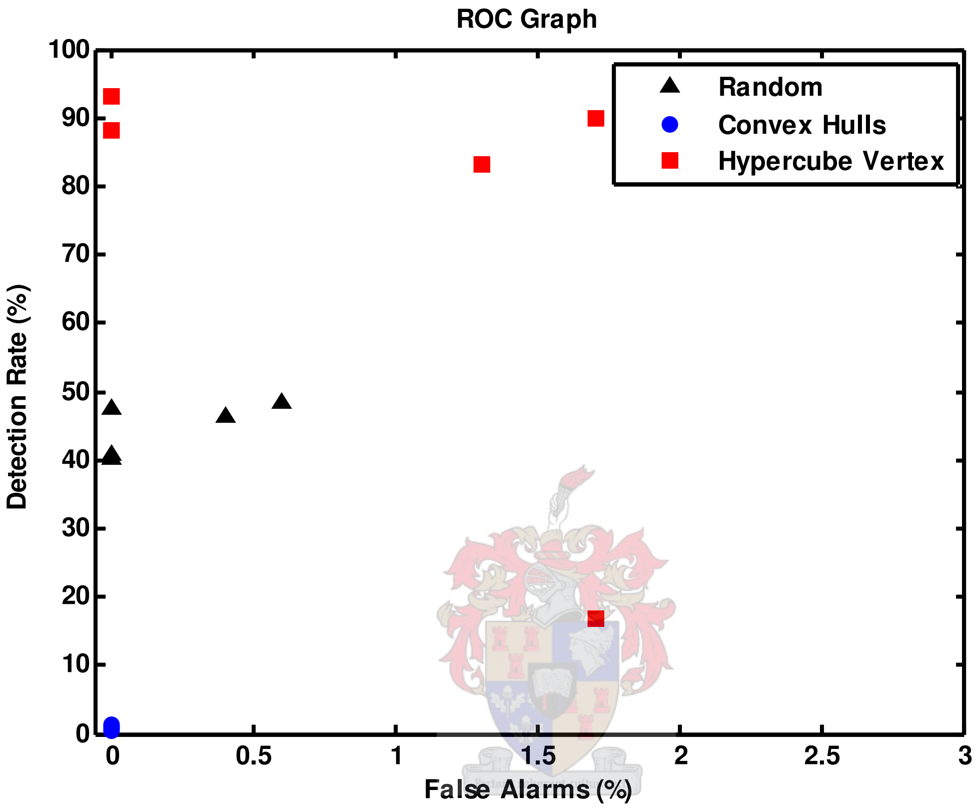


Figure 48: A combined ROC graph comparing the performance of three different methods to generate detectors for the autocatalytic reaction.

From Figure 48 it seems though the hypercube vertex method once again surpasses the other methods in both detection capability and false alarm minimisation. It does not only exhibit the highest scoring individual run, but it also provides a higher scoring average, which means that if the parameters are accidentally miss-chosen the hypercube vertex method would still provide an improved classification. The convex hulls method is unsuccessful in its classification. With almost no detection capability and the highest false alarm rate it is not recommended for use in monitoring the autocatalytic reaction. Random detectors may provide some classification if parameters are properly defined.

Chapter 6: Applying AIS: Belousov-Zhabotinsky Reaction

6.1 Introduction

The final case study involves the Belousov-Zhabotinsky reaction. The Belousov-Zhabotinsky (BZ) reaction is defined as a chemically active medium, which maintains self-oscillations and propagating waves (Lu and Teulilo). Originally, the BZ reaction involved the oxidation of malonic acid by bromate ions (Potassium Bromate) in a sulphuric acid medium. Cerium was used as a catalyst in the original reaction, which presents a pale yellow in the Ce^{4+} state and clear in the reduced Ce^{3+} state. The oscillations in the reaction can be observed by means of a rapid oscillating colour change in the liquid. The reactions that describe the BZ reaction are given below:

$$\begin{array}{ll} A + Y \rightarrow X + P & r = k_3 AY \\ X + Y \rightarrow 2P & r = k_2 XY \\ A + X \rightarrow 2X + 2Z & r = k_5 AX \\ 2X \rightarrow A + P & r = k_4 X^2 \\ B + Z \rightarrow \frac{1}{2} fY & r = k_c BZ \end{array} \qquad [22\ \text{a-e}]$$

Where:
$A=HBrO_3^-$
B: All oxidizable organic species
$P=HOBr$
$X=HBrO_2$
$Y=Br^-$
$Z=M_{ox}$

f is a stoichiometric factor, which relates quantitatively between the reactants and the products. The stoichiometric factor requires a partial knowledge of the chemistry that is involved. In this scenario, f is chosen as 1, but it can typically range between values of 1.25 and 0.25.

The reaction rate equations for the intermediate species X, Y and Z are:

$$\frac{dX}{dt} = k_3AY - k_2XY + k_5AX - 2k_4X^2$$
$$\frac{dY}{dt} = -k_3AY - k_2XY + \frac{1}{2}fk_cBZ \qquad [23\ a\text{-}c]$$
$$\frac{dZ}{dt} = 2k_5AX - k_cBZ$$

To obtain dimensionless equations, the following dimensionless variables are required:

$$x = \frac{2k_4X}{k_5A} \qquad y = \frac{k_2Y}{k_5A} \qquad z = \frac{k_ck_4BZ}{(k_5A)^2}$$
$$c_1 = \frac{k_cB}{k_5A} \qquad c_2 = \frac{2k_ck_4B}{k_2k_5A} \qquad q = \frac{k_3k_4}{k_2k_5} \qquad [24\ a\text{-}c]$$
$$\tau = k_cBt$$

The dimensionless differential equations can now be formulated which will govern the model:

$$\frac{dx}{d\tau} = \frac{1}{c_1}[qx - xy + x(1-x)]$$
$$\frac{dy}{d\tau} = \frac{1}{c_2}[-qy - xy + fz] \qquad [25\ a\text{-}c]$$
$$\frac{dz}{d\tau} = x - z$$

According to (Lu and Teulilo, 2002), typical values for c_1, c_2 and q are:

- $c_1 = 0.04$
- $c_2 = 0.0002$
- $q = 0.0008$

There are three main processes in the BZ model: The bromide consumption stage, the autocatalytic stage and regeneration stage. The Bromide transforms into different oxidized forms, denoted by process A. The role of this process is to remove the Br^- ions. This ion is the inhibitor for process B (autocatalytic stage), whilst $HBrO^2$ is the activator. Process B only starts when the bromide ion reaches a lower critical value. The oxidation of the catalyst transforms Ce^{3+} into Ce^{4+} (Lu and Teulilo, 2002).

With the model defined and normal data generated, the anomalous data is generated by allowing q to drift, which implies a change in the rate constants. In 10 increments, q is increased by 8E-5 for each increment. This causes q to reach a value of 0.016 after the drift. Figure 49 shows a section of both the normal data and the second steady state anomalous data as obtained from the Runge-Kutta method. The mid-section, i.e. during the drift, is once again not shown for practical reasons.

The difference between normal and anomalous data does not seem to be significant, possibly excluding for y, on which q has a larger influence, due to a smaller c_2. Once all data is generated, the transition between measurement space and shape space may be initiated.

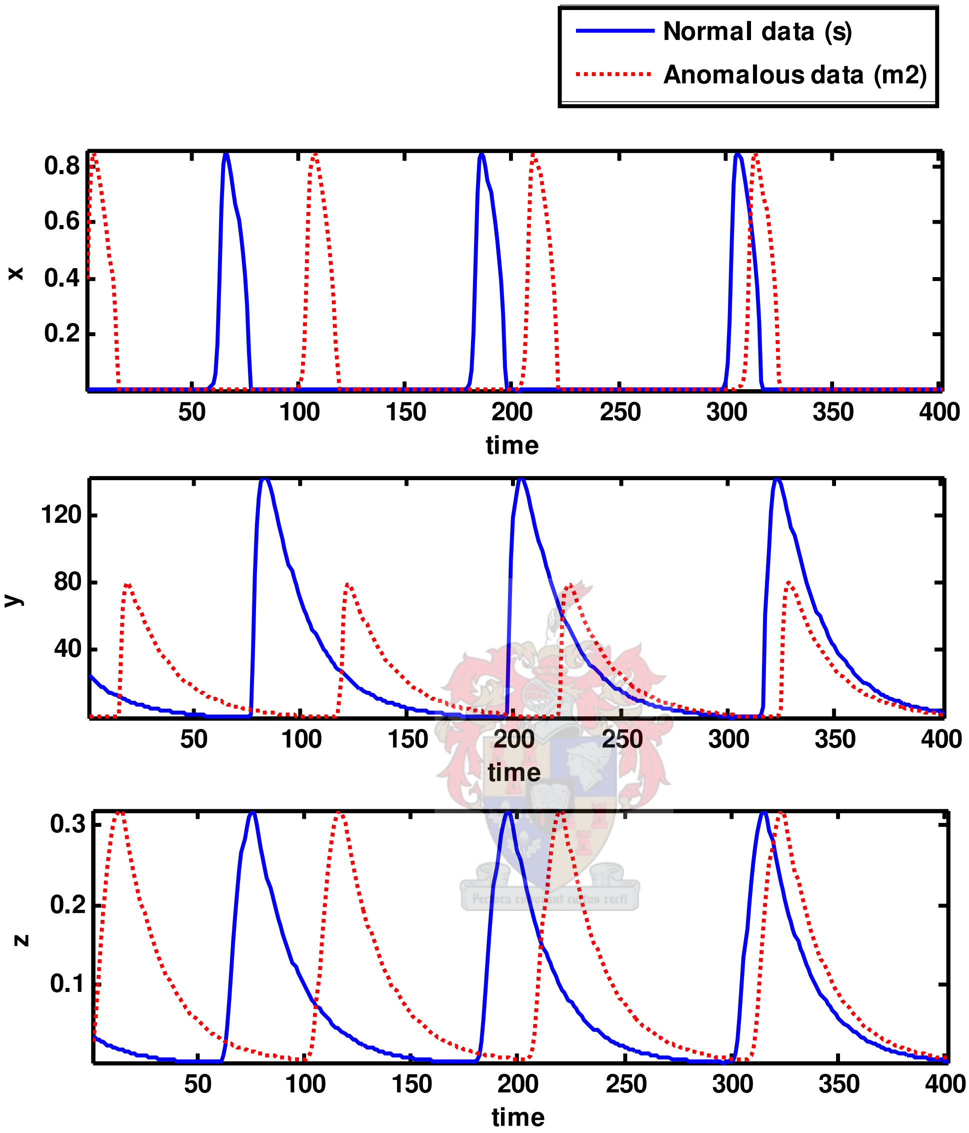


Figure 49: Representation of the BZ reaction equilibrium: The first steady state is indicated in a solid line and represents normal process behaviour; the second steady state is shown in a dotted line and represents anomalous behaviour.

6.2 Transforming between measurement-space and shape-space

Figure 50 shows the correlation function for the BZ data. From the correlation function a lag of 12 measurements provides an uncorrelated self space.

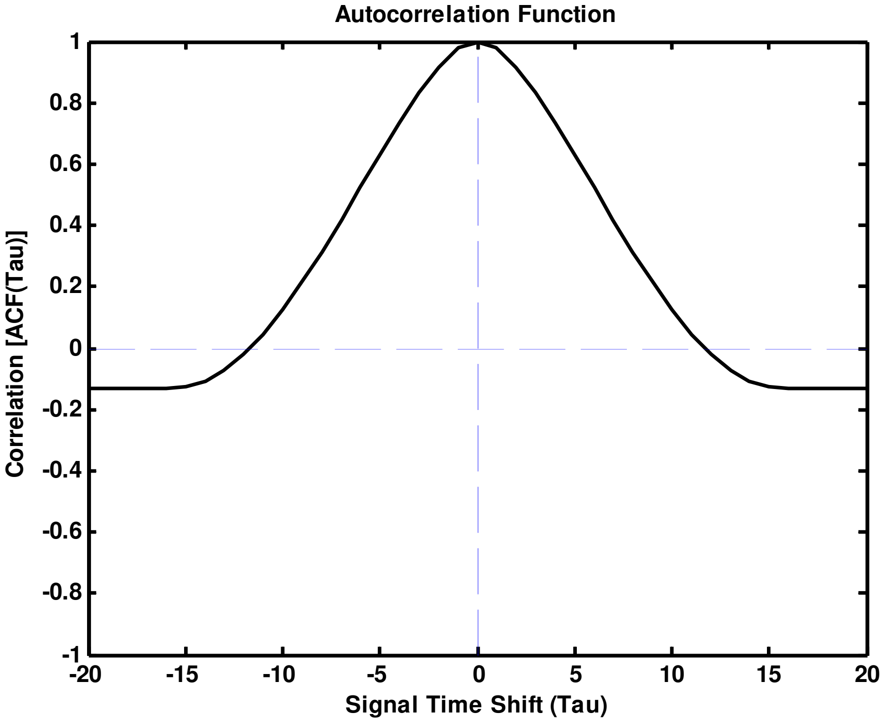


Figure 50: Autocorrelation function of time series data s for the BZ reaction.

Prior to applying PCA, the time series are mean centred and scaled. The 36 dimensional matrix [12 (lag) x 3 (x,y and z)] is reduced to 3 dimensions. This decision is justified by considering the cumulative sum of eigenvalues, refer to Figure 51. While 3 dimensions capture 96.4% of the data variation, it also enables one to view the data graphically, whereas for example 4 dimensions would disallow this feature.

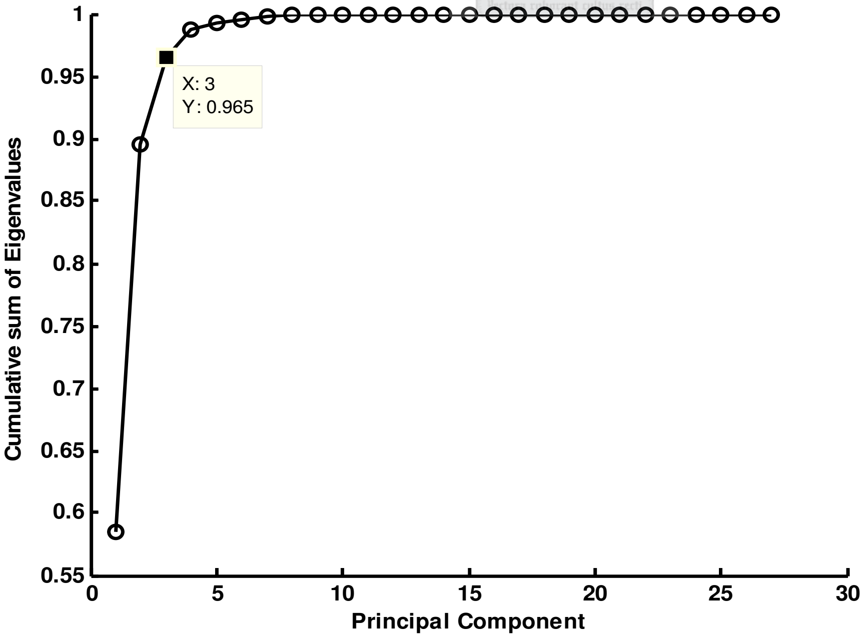


Figure 51: Cumulative sum of latencies of the principal components of self space S of the BZ reaction data.

Figure 52 shows the 3 dimensional shape space of the BZ reaction. It too has several possible holes in the shape space, which will influence the value of ε in the same way as is the case in the Predator-Prey case study.

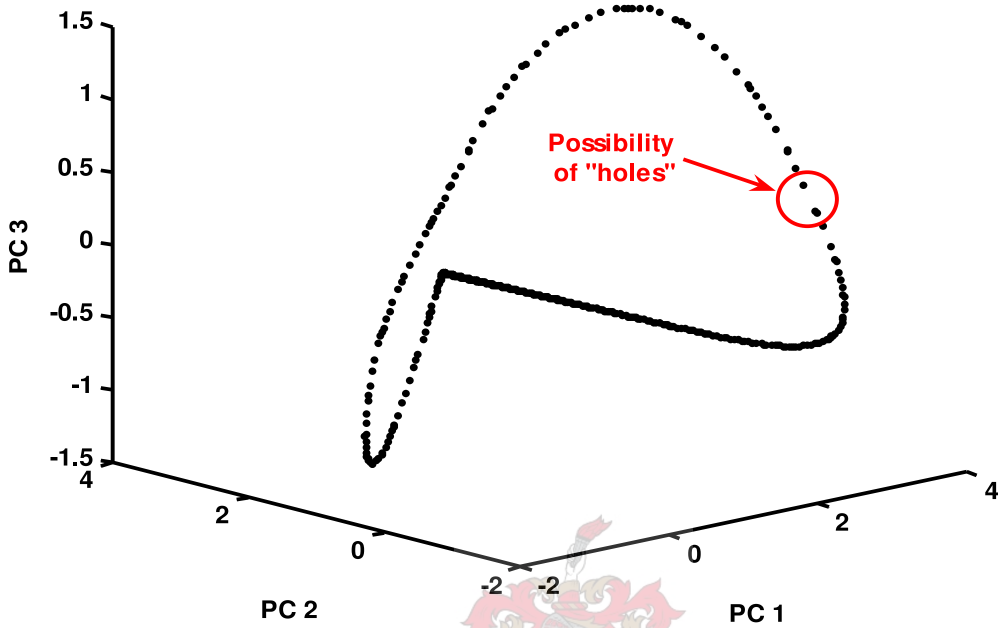


Figure 52: 3-Dimensional Principal Components plot of the Self-Space of the autocatalytic reaction data.

With the self space constructed, the detectors may be generated. The various methods will once again be investigated and their performances compared.

6.3 Generating Detectors and Classifying Data

In the previous two case studies, the hypercube vertex method is recommended for both its reliability and repeatability. Random detectors, though the best known method in literature is not the best method in any of the previous case studies. Another case study is now considered to determine the accuracy of the ranking in methods for generating detectors.

6.3.1 Randomly Generated Detectors

The first task, as discussed in previous chapters, in generating random detectors is determining the detector affinity threshold ε. This is once again calculated by considering the maximum distance between neighbouring events in the self space, as discussed in Chapter 3.

After ε has been determined, the number of detectors can be calculated. This is done by estimating the detector coverage as a function of the number of detectors. Figure 53 shows both time needed to generate x detectors and the estimated coverage that x detectors will provide.

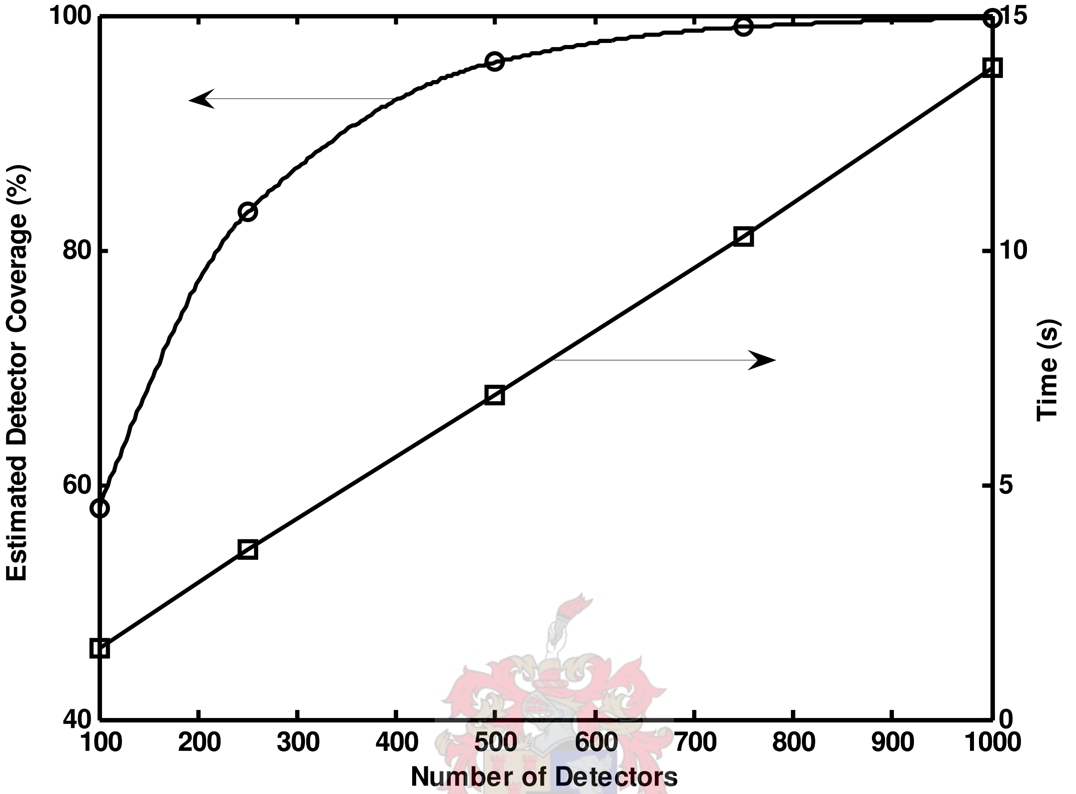


Figure 53: Estimated random detector coverage and time taken to generate these detectors as a function of the number of detectors for ε = 0.5.

Figure 53 shows a linear increase in the time required to generate a given number of detectors. A large increase (14.8%) in estimated detector coverage is observed from 250 to 500 detectors accompanied by a 3.3 second rise in computational time. Between 500 and 750 detectors however, a small increase (3%) is seen for a 3.4 second increase in computation time. It seems unfeasible to generate 750 detectors at double the cost, when 500 will be adequate.

Figure 54 shows the complete shape space with self, nonself, detectors and identified anomalies. At first glance it appears as if the random detectors are unable to classify the data correctly. This is investigated in detail with results summarised in Table 9.

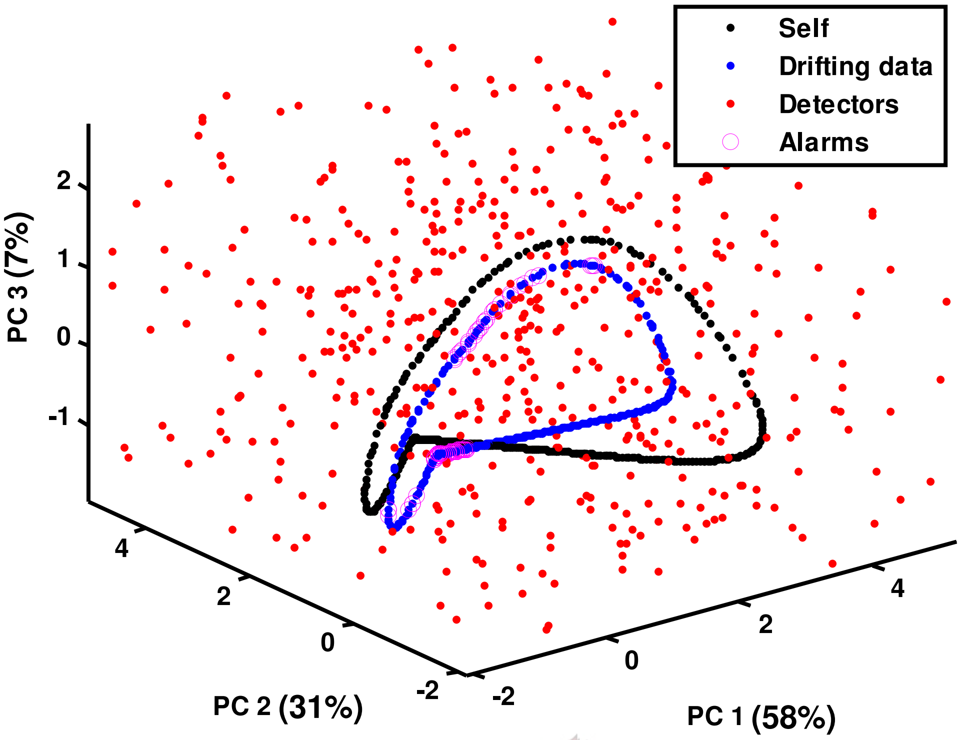


Figure 54: 3-Dimensional presentation of 500 random detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Table 9 substantiates the speculation that random detectors are not adequate in classifying the BZ data. Though no false alarms occur, merely 49% of the true positives are identified. The performance of the random detectors in monitoring the BZ data is better seen when considering the ROC graph (Figure 55).

Table 9: Summary of AIS detection capability for the BZ reaction data, monitored by 500 randomly generated detectors.

Run	ε	Number of detectors	Alarms Triggered		
			Validation (s)	Drifting (m1)	Drifted (m2)
1	0.2	2000	0.0%	6.5%	9.0%
2	0.3	900	0.0%	7.2%	13.0%
3	*0.4*	*600*	*0.0%*	*16.3%*	*17.6%*
4	0.5	500	0.0%	25.0%	49.1%

Run 3, shown in *italic* is the chosen optimum setup.

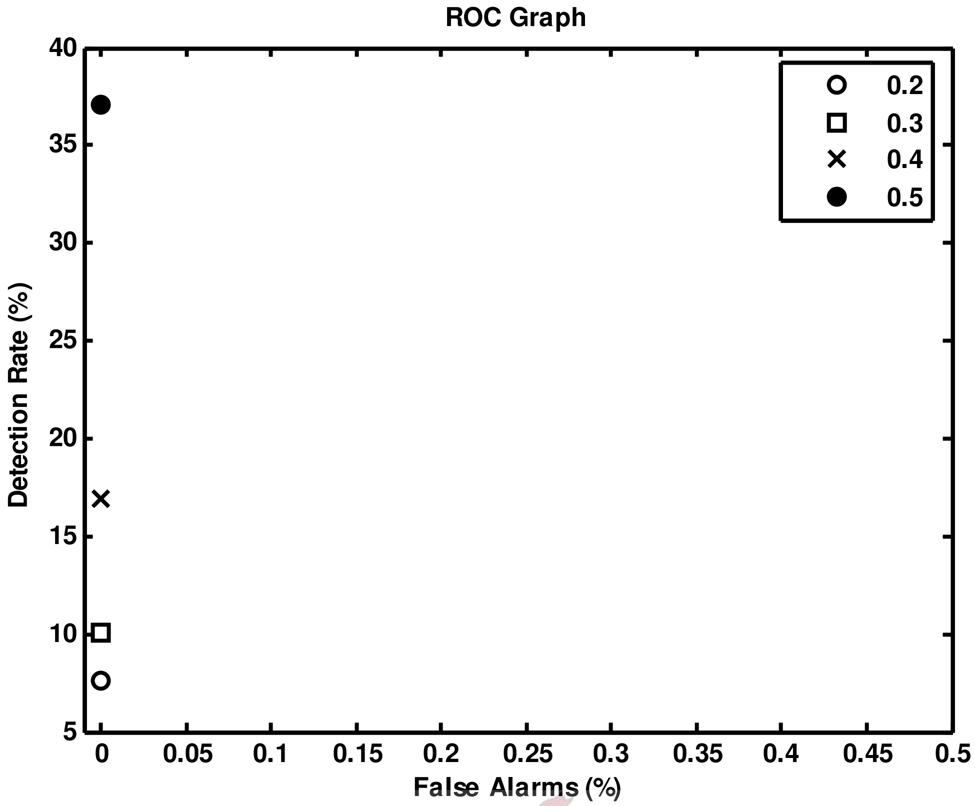


Figure 55: The ROC graph, comparing various different values for ε during random generation.

6.3.2 Convex Hulls

The theory behind the convex hulls approach is discussed in Chapter 3, while the application thereof is described in Chapters 4 and 5. The method is once again applied, but this time to the BZ reaction data. Uniformly distributed random noise of magnitude ε is added to the self space to avoid autoimmunity. A detector set of unfixed size is generated for every run on which noise is added. This process is repeated until the detector set reaches a predetermined size. Between runs, the current noise is removed and new random noise is added, causing detectors to be placed on different pseudorandom locations. The size of the final detector set is determined by estimating the detector coverage, as discussed in Chapter 3. Figure 56 shows the estimation of coverage and computational time as a function of the number of detectors. The required time varies between 0.07s and 0.09s and can be deemed negligible, as 0.02s is imperceptible. In the range of 250 – 1000 detectors, all provide reasonable coverage and any number may therefore be chosen. For comparative reasons however, 500 detectors are once again chosen.

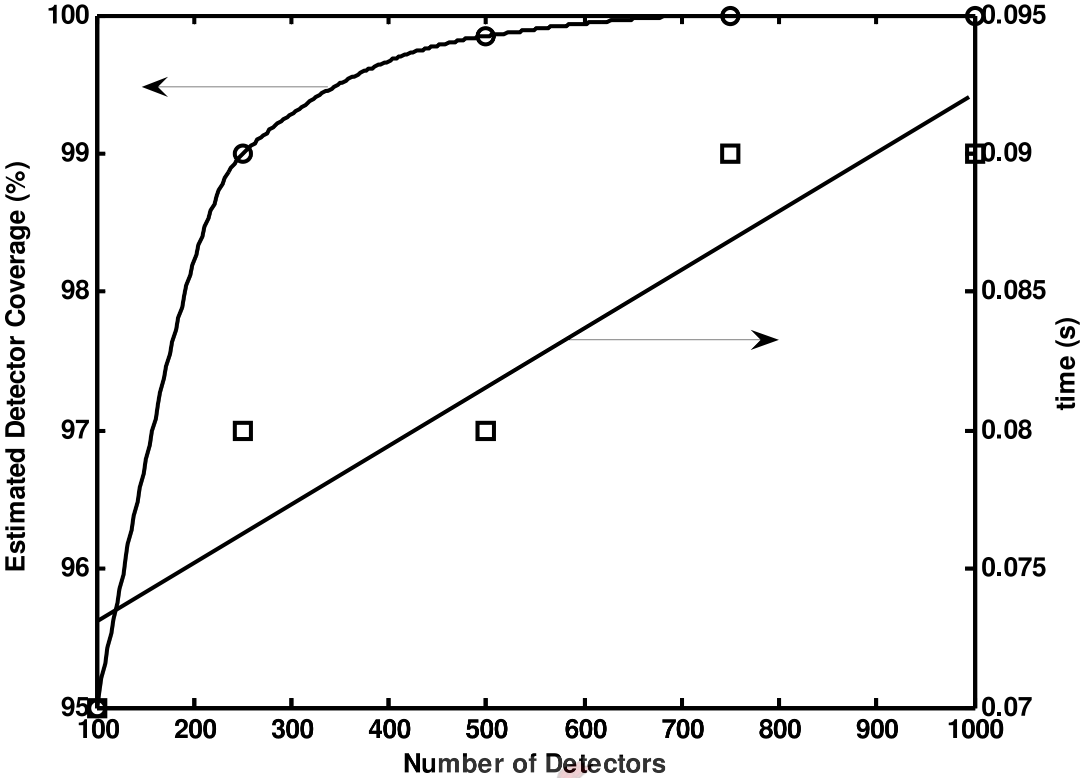


Figure 56: Estimated convex hull detector coverage and time taken to generate these detectors as a function of the number of detectors for ε = 0.5.

With the number of detectors chosen, the shape space may be presented (Figure 57). The detectors are aligned around the self space, but almost no anomalies are detected. This is due to the nature of the convex hulls around the self space. Detectors are not misplaced, but they do not cover all the areas around the self space, even though their estimated coverage is near 100%. The coverage is estimated by counting the number of detectors that overlap in their affinity threshold, ε. If all detectors overlap, it is estimated that 100% coverage is achieved. This is true for the narrow band in which the detectors are placed, but the band itself does not reach far enough to cover the nonself space. In the case of random detectors, the estimated coverage is fairly accurate, because detectors tend not to cluster together. In the case of convex hulls and even hypercube vertex detectors, the detectors are generated in more or less the same area of space. This will cause more of the detectors to overlap and at the same time, more open spaces in the nonself space. The hypercube vertex method is better equipped in the sense that not only two bands (inner and outer), but 2xn bands, where n is the number of dimensions. Hence, in 3 dimensions, 6 bands of detectors are formed, contrasted to of the two bands in convex hulls. They are also more strategically placed, as opposed to random noise being added around the self space. Full elaboration of this concept will follow in later sections.

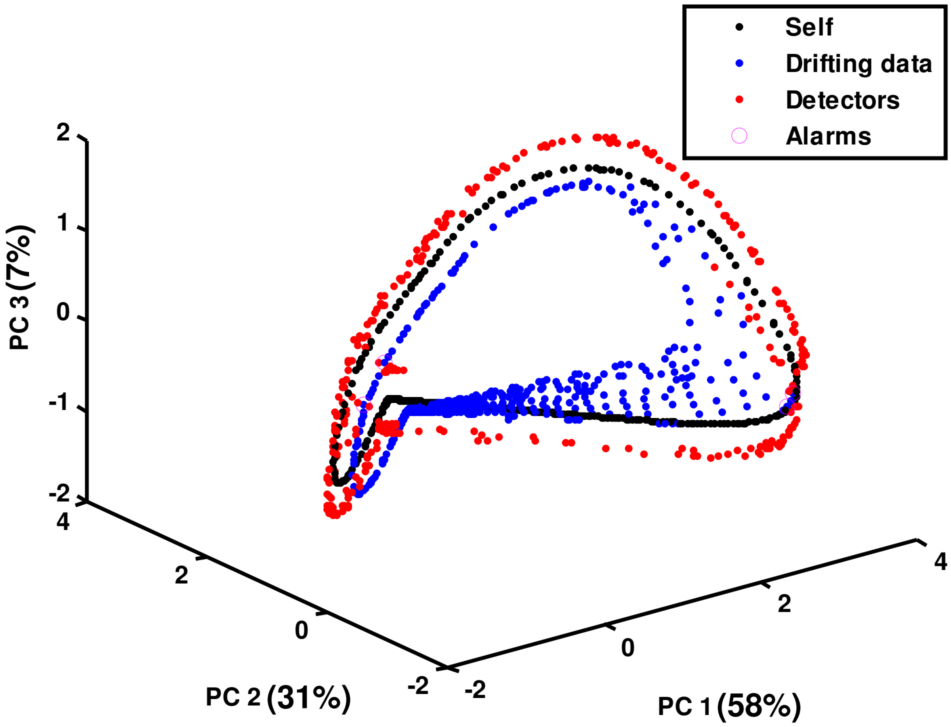


Figure 57: 3-Dimensional presentation of 500 convex hulls detectors around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Though convex hulls are clearly incapable of detecting change in the BZ reaction, ε is still varied to complete the study and fully compare results in all scenarios. Table 10 shows the discouraging results of this study, which are better visualised by a ROC graph, shown in Table 10. the optimum setup for this scenario is ε = 0.4.

Table 10: Summary of AIS detection capability for the BZ reaction data, monitored by 500 convex hulls detectors.

Run	ε	Alarms Triggered		
		Validation (s)	Drifting (m1)	Drifted (m2)
1	0.1	0.5%	0.7%	0.0%
2	0.2	2.7%	1.6%	0.0%
3	0.3	2.7%	2.7%	1.4%
4	*0.4*	*1.1%*	*1.6%*	*1.6%*
5	0.5	1.1%	3.6%	4.7%

Thus far, no method was successful in detection change in the BZ reaction. However, the current method of choice, the Hypercube Vertex method is yet to be tested in this system. The following section therefore considers the hypercube vertex method.

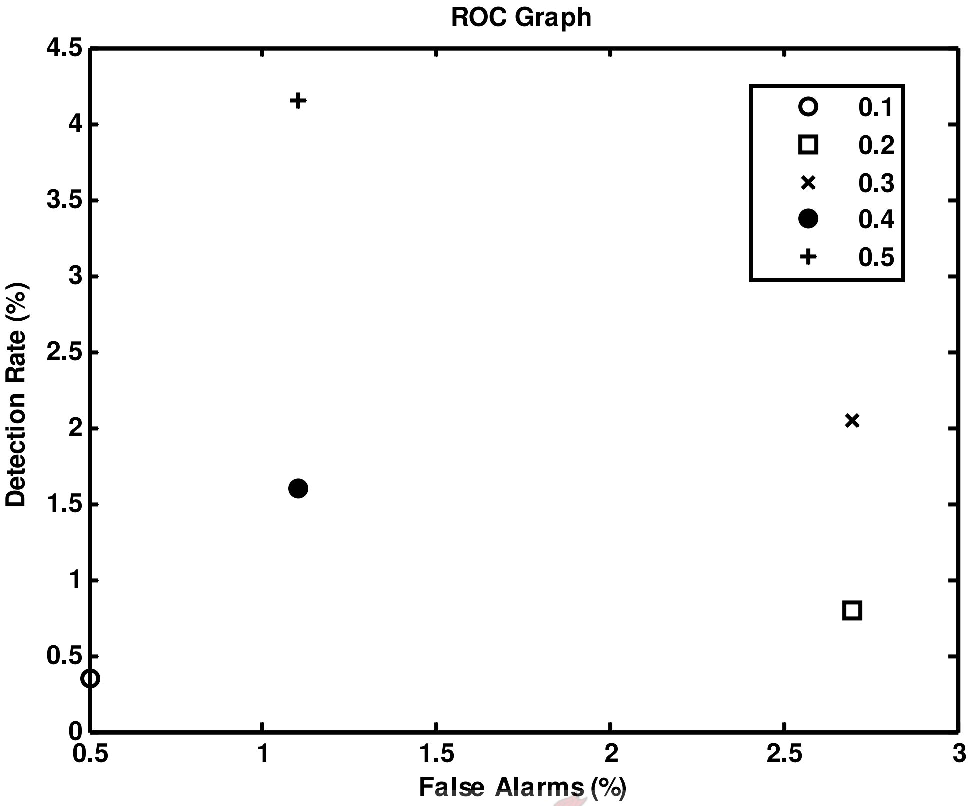


Figure 58: The ROC graph, comparing various different values for ε during random generation.

6.3.3 Hypercube Vertex

The hypercube vertex method is expectedly better equipped to monitor the BZ reaction. However, in order to compare the various methods, the number of detectors generated by the hypercube vertex method will have to be limited to 500. This is done by altering the generating interval (n), which means that every nth point in the self space will be used to generate detectors. Figure 59 shows that for n=7, 427 detectors will be generated. Although this figure is less than 500, choosing n=5 would result in 586 generated detectors. By choosing n=7, the error will be minimised and a standard for comparison between the various methods is still ensured. Figure 60 shows the shape space with the 427 detectors and anomalous data identified. At first glance the performance of the hypercube vertex method seemingly surpasses that of any method thus far examined. Its true performance can however only be judged when ε has been varied to test the robustness of the method. The results of this test are shown in Table 11, whilst a graphical illustration thereof is presented by means of a ROC graph in Figure 61.

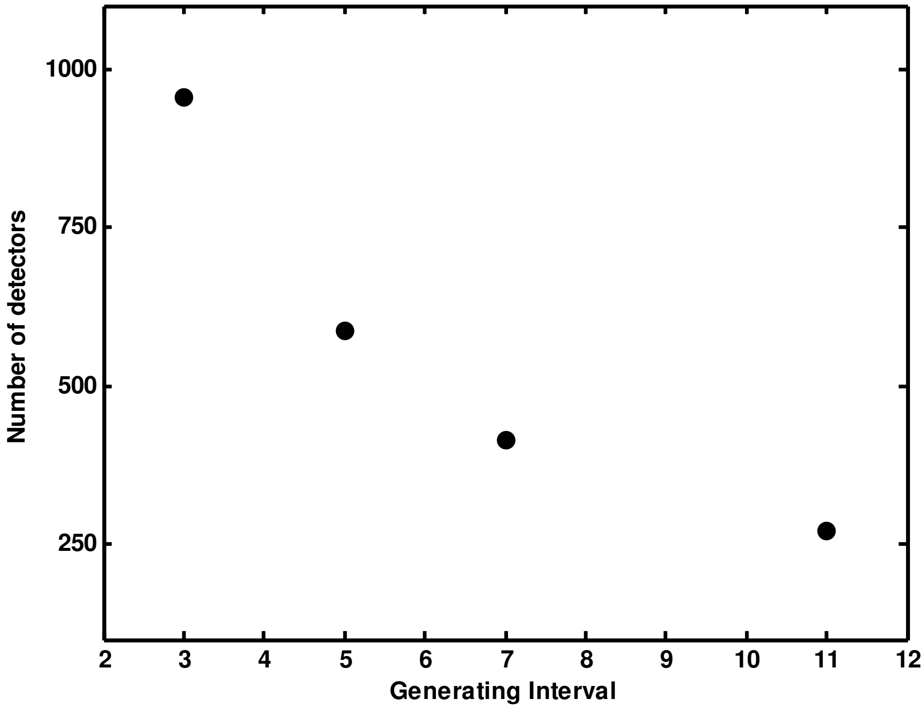


Figure 59: Number of detectors generated using the Hypercube Vertex method versus generating interval.

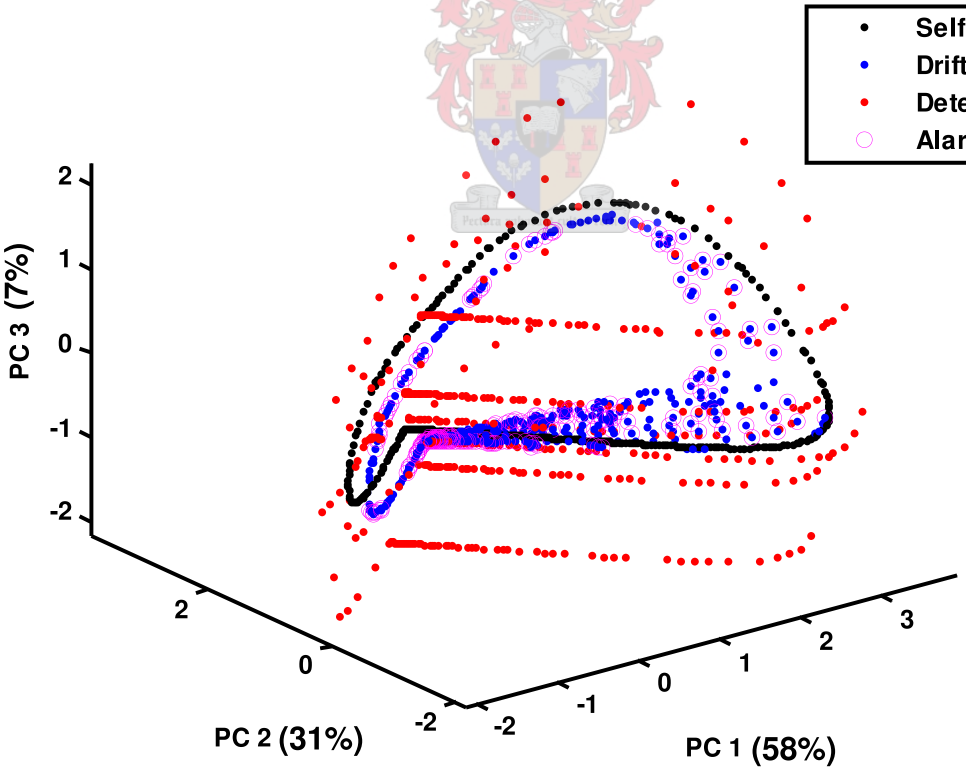


Figure 60: 3-Dimensional presentation of 427 Hypercube Vertex detectors (with ε = 0.2) around the self space. Anomalous data are also shown and those identified as anomalous are encircled.

Table 11: Summary of AIS detection capability for the Predator-Prey data, monitored by 427 Hypercube Vertex detectors.

Run	ε	Alarms Triggered		
		Validation (s)	Drifting (m1)	Drifted (m2)
1	0.1	3.2%	69.0%	66.7%
2	*0.2*	*1.1%*	*70.7%*	*78.9%*
3	0.3	3.2%	62.5%	80.3%
4	0.4	0.5%	50.5%	75.0%
5	0.5	0.0%	27.0%	65.9%

According to the ROC graph in Figure 61, the best configuration is found when ε = 0.2. This is advocated by the estimated value of ε as discussed in Chapter 3. The furthest distance calculated between neighbouring events in the self space is 0.16. Upon adding a safety factor to accommodate for the holes, it can easily be seen that the optimal value for ε is 0.2. The safety factor used does, however, seem flexible for different case studies and different methods used. The calculated value for ε only provides a guide towards the optimal value and it is recommended that the value be tweaked for best results.

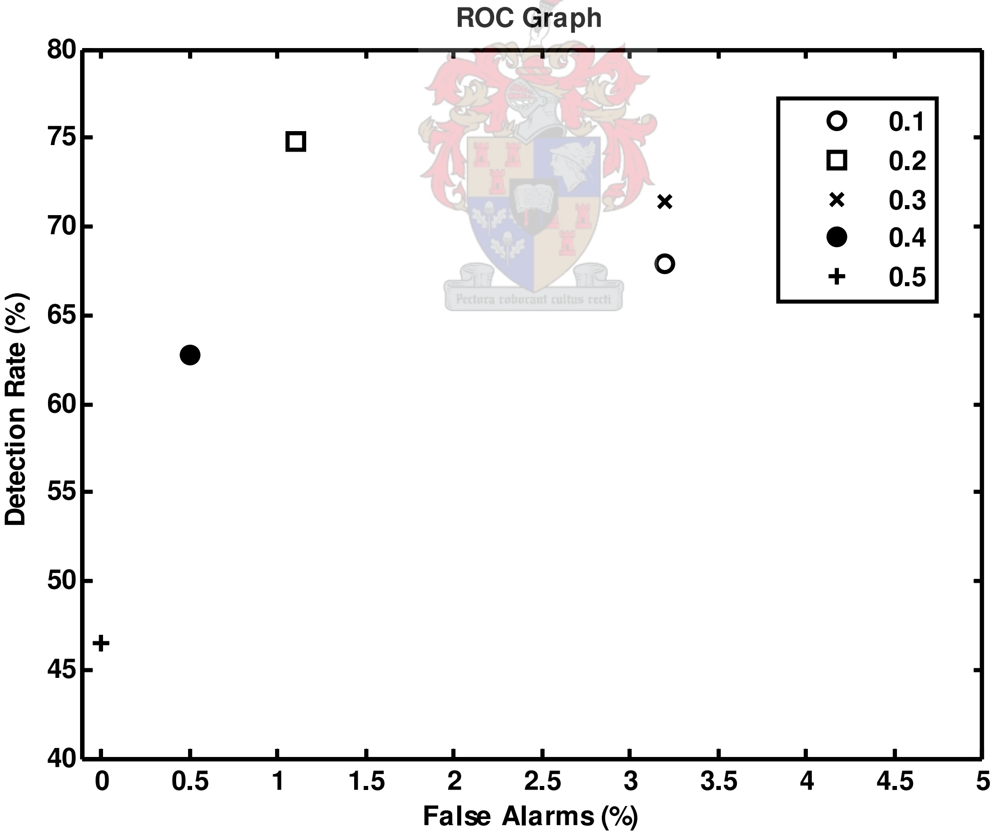


Figure 61: The ROC graph, comparing various different values for ε during the Hypercube Vertex Approach.

6.4 2-Dimensional Shape Space

This section provides a visual comparison of the detector coverage for various values of the detector affinity threshold ε; refer to Figure 62.

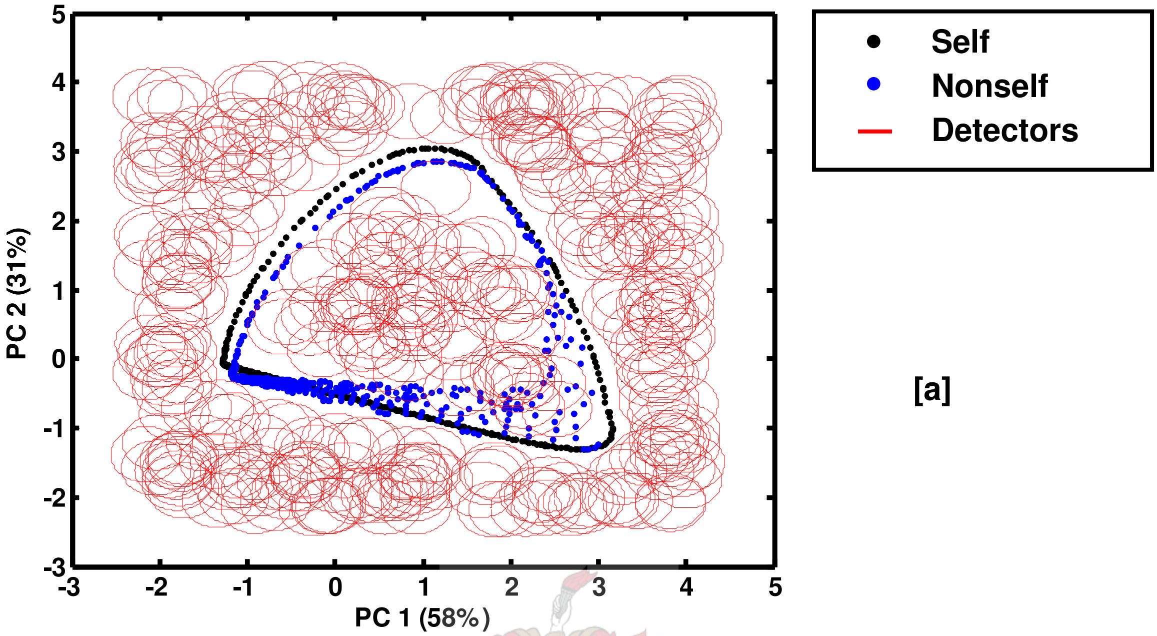


Figure 62 [a]: 2-Dimensional Shape Space of the BZ-reaction data with 500 random detectors and ε = 0.5.

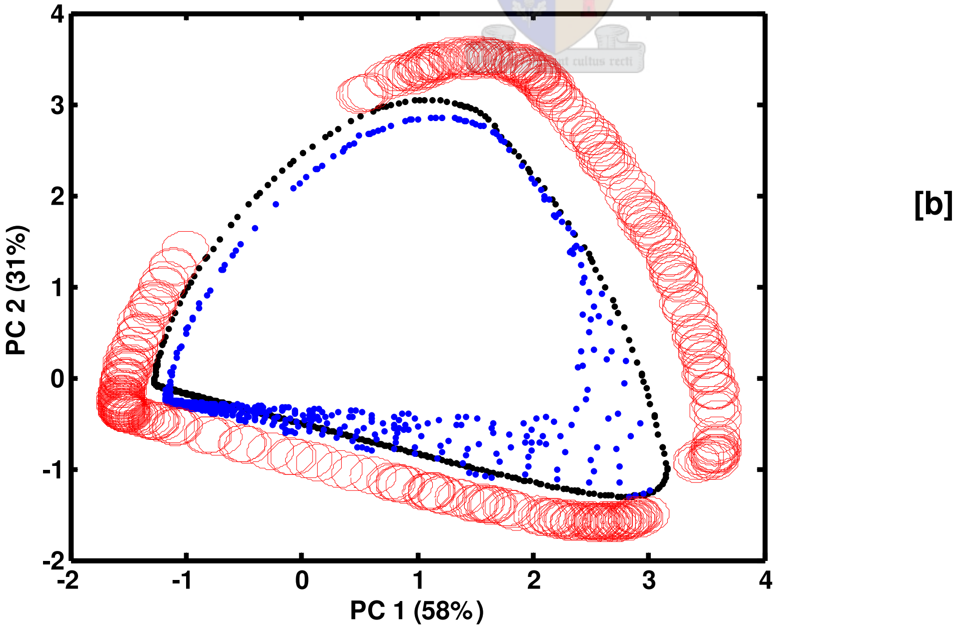


Figure 62 [b]: 2-Dimensional Shape Space of the BZ-reaction data with 500 convex hulls detectors and ε = 0.5

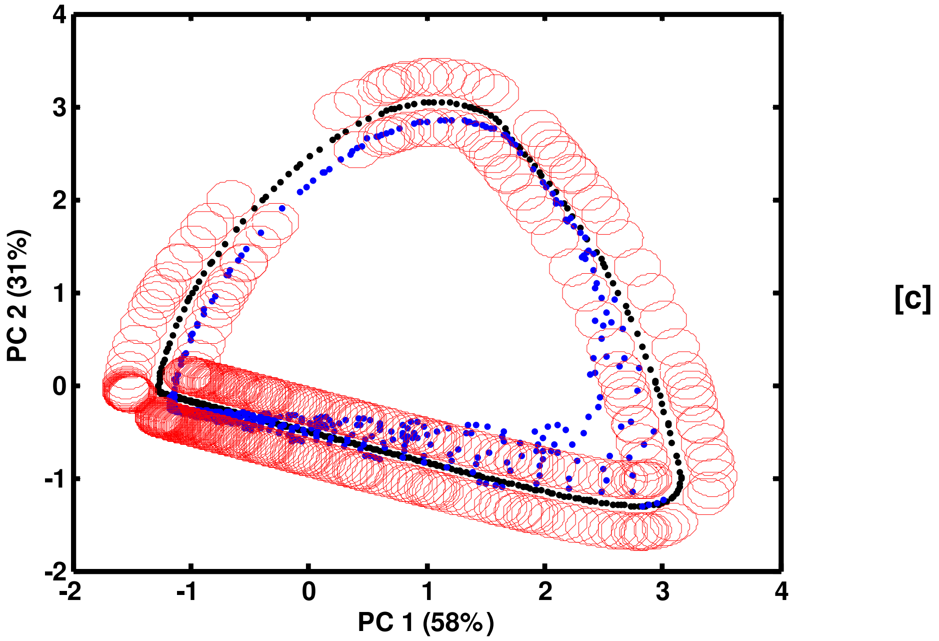


Figure 62: 2-Dimensional Shape Space of the BZ-reaction data with 427 hypercube vertex detectors with ε = 0.2.

Gaps in the detector coverage are present in both the convex hulls and hypercube vertex methods. It should again be noted that only two reasons exist for detectors not being located in positions that one would expect them to be. These are (a) no detectors were placed here in the first place or (b) the detectors originally placed here were subsequently removed in the screening process. In convex hulls, the gaps are explained by drawing an imaginary connection covering the gaps in the detectors' trajectories. It is then noticed that the detectors falling in this trajectory would overlap with the self events in the self space and were therefore removed by the system. The reason for the gaps in the convex hulls scenario is therefore reason (b) as stated above. In the hypercube vertex scenario, inspection of the two detector trajectories (inner and outer) reveals that both trajectories are missing detectors in the same areas. It is therefore, less likely that the detectors have been screened. More possible, is that they have never been placed in these locations in the first place. A reason for this is probably that this section of the trajectory was skipped on every pass due to the fact that only every nth event in the self space is used for generation of detectors.

6.5 Effects of Noise in the Belousov-Zhabotinsky reaction

The average detection rate in the Belousov-Zhabotinsky reaction is lower than the other systems for all methods of generating detectors. The convex hulls method again exhibits lacking results. The nature of the shape space of the BZ reaction is such that the self space and nonself space overlap to a larger extent than any other case study. This is shown in Figure 63. In this section, noise is added to the BZ-reaction data at 10% of the standard deviation in the measurement space.

The first three principal components maintain 96% of the data variation.

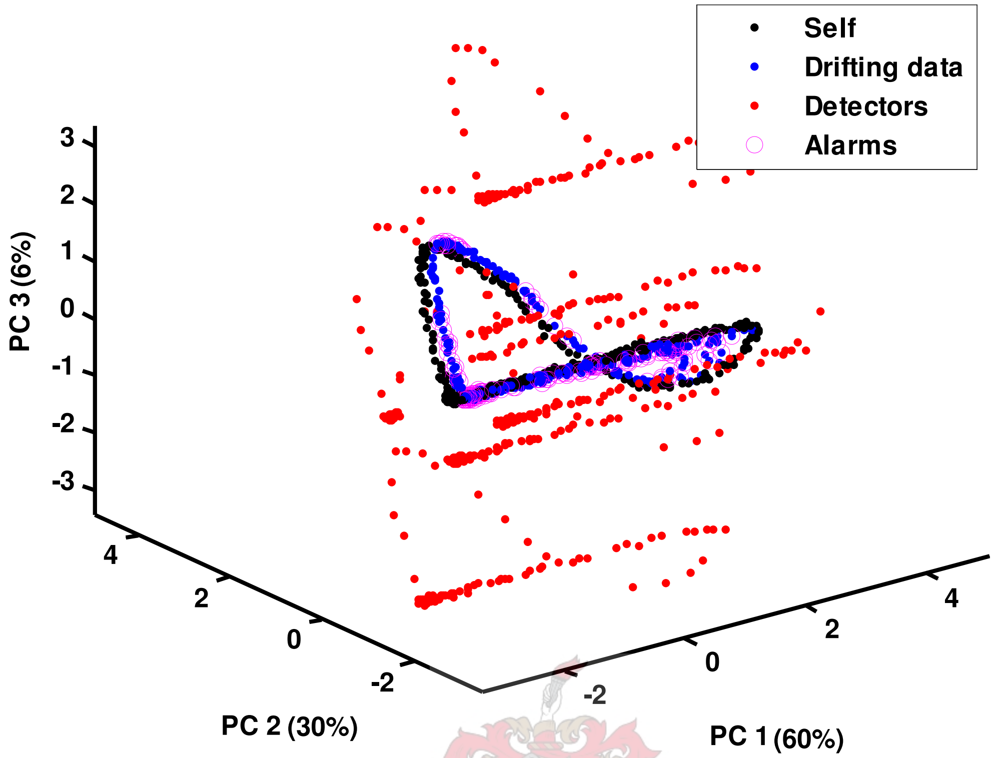


Figure 63: The shape space of the BZ reaction with noise added at 10% of the standard deviation.

Because of the significant overlap between self space and nonself space, the classification is deficient. In the drifting data, 45% of the anomalous events are correctly classified. The second steady state (m2) however, is classified to an accuracy of 76%, which gives an average detection of 60.5%. A false alarm rate of 0.5% is achieved throughout.

6.6 The Computational Cost of the Various Methods

This section investigates the computational cost evolved in generating the various types of detectors. Other phases of the AIS such as creating the self space and monitoring the process are constant between different methods for generating detectors due to unchanging methodology. Table 12 summarises the computational time involved with generating one detector of a given method.

Table 12: Computational cost associated with the various methods for generating detectors

Case Study	Time per detector (ms/detector)		
	Random	Convex	Hypercube
Predator-Prey	21.6	1.8	30.0
Autocatalytic reaction	17.3	2.3	24.8
BZ Reaction	13.8	1.6	25.6

The requirements of the Hypercube Vertex detectors surpass that of any other method. Their reliability, however, far outweigh the cost they entail. The convex hulls detectors demand the least computational strain of all detectors but are also the most unreliable. In short, the additional cost involved in generating hypercube vertex detectors is exceeded by their superior ability to detect change without raising unnecessary false alarms.

6.7 Summary of Chapter 6

The oscillating Belousov-Zhabotinsky reaction is introduced in this chapter. The reactions are given and the differential equations that model the reaction is presented. Figure 49 shows a section of the data illustrating the behaviour of the BZ reaction. Thereafter, the chapter proceeds by explaining the transformation between measurement space and shape space. The autocorrelation function is used to determine the initial dimensionality of the shape space, where after the dimensionality is reduced by means of Principal Component Analysis. The latencies are taken into consideration when deciding on a 3 dimensional shape space, which retains 96.4% of the original data trends. Calculating the detector coverage suggests that 500 detectors adequately cover the nonself space. The detector affinity threshold ε was estimated at 0.16, which was altered to test for the optimum value. Figure 64 shows a combined ROC graph which describes the performance of the different methods.

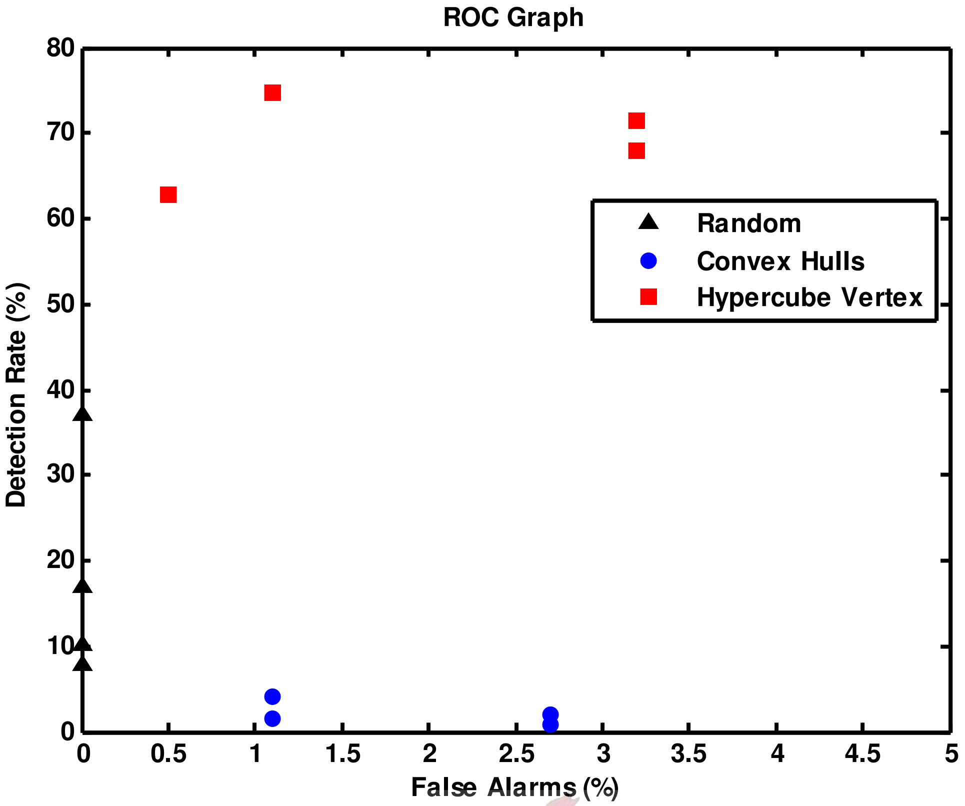


Figure 64: A combined ROC graph comparing the performance of three different methods to generate detectors for the BZ reaction data.

With the Hypercube Vertex method having 15% less detectors than the other methods, it still excels in both detection rate and false alarm rate. This is true not only for a single value of ε, but also the average reliability of the method given an arbitrary value for ε, within acceptable bounds.

The chapter concludes by comparing the computational cost involved in generating a single detector with a specific method. It finds that the hypercube vertex method takes the most time to generate a single detector, while the convex hulls detectors are generated the fastest. The time involved in generating detectors are however measured in milliseconds and is therefore not a major factor in the decision hierarchy.

Chapter 7: Conclusions

In the conclusions, Artificial Immune Systems are weighed on the comparative criteria given in Chapter 1. This provides an understanding of the performance of AIS in comparison to existing methods. A summary is given on the performance of AIS in the various case studies and mention is made to their performance under noise. Some comments are made to the computational cost analysis of generating detectors, which is computationally the most taxing stage, though this was not formally investigated.

The 10 criteria on which fault diagnostic and identification systems are measured are:

Fast Detection and Diagnosis:

The measure of how fast AIS detect change can be said to relate to the percentage detection rate in the drifting phase of the case studies. In the Volterra process, 100% of drifting data were identified, which indicates that even the slightest anomalous tendency is identified. Even with noise only 4% of the drifting data are not identified as anomalous. It can therefore be said that AIS indeed respond quickly to process faults.

Isolability:

At this stage, isolability of process faults are yet to be achieved using AIS. The use of a shape space that is formed using Principal Component Analysis somewhat inhibits the ability of this fault detection system to isolate the origin of the anomaly. This is because there is no direct correlation between the measured variables and the principal components. Future work in this field may provide an answer to this problem.

Robustness:

Chapter 7 shows that AIS are robust to noise with tolerable decrease in detection rate which translates to swiftness in response. AIS can therefore, be characterized as robust without unreasonable loss of rapid response. More work is however, required on this topic to provide elaboration.

Novelty Identifiability:

The design of AIS is based on Novelty Identifiability. The system uses the unique working of the natural immune system to identify anomalous process behaviour by at the same time avoiding autoimmunity.

Classification Error Estimate:

The estimate of classification error is given throughout the thesis, with values ranging between 0% and 1%. These values are obtained by considering the rate at which false alarms are generated in the validation stage. The classification error estimate is therefore available to the user prior to the classification process or monitoring phase.

These figures are only applicable in the case studies considered in this thesis, but do provide a ballpark figure for other dynamic systems.

Adaptability:
An Artificial immune system can generate a new set of detectors at predetermined intervals. A progressive understanding of the process at hand is therefore shaped. Changing inputs or equipment ware can be accommodated for and no preventable false alarms will thus occur. The hypermutation algorithm would, for example be able to mutate and adapt to any alteration to the system.

Explanation Facility:
The ability to reason about causality is beyond the scope of AIS. The natural immune system is unaware of the cause of an infection of disease; it simply identifies the problem and takes appropriate action.

Modelling Requirements:
Since AIS are quantitative methods, modelling a specific process is not required when initiating AIS. Certain preprocessing of historic data is however required.

Storage and Computational Requirements:
AIS are real time systems and have algorithms that are computationally undemanding. They're storage requirements are also low. In the case studies considered for example, around 500 detectors are adequate to monitor the processes. These detectors occupy hard disk space in the order of kilobytes.

Multiple Fault Identifiability:
Any fault, be it the interaction between various faults or a previously unknown fault, will produce an event in the shape space that falls outside the self space. This will trigger an alarm from a nearby detector, provided the detectors are adequately placed.

Table 13 summarises the criteria above, also introduced in Table 1 with the exception of the final column giving a summary of the capabilities of AIS.

A discussion regarding the natural immune system and its relevance towards artificial immune systems is given in Chapter 2. Four different artificial immune algorithms are discussed, with the negative selection algorithm emerging as the method of choice for this particular project. The hypermutation method is mentioned in Chapter 2, but it is stated that this method, despite its popularity in the literature, is not capable of meeting the requirements in this project. Its main drawback is that it requires knowledge of the nonself space in order to mutate and produce better performing detectors. Various applications of the negative selection algorithm are provided, while it is mentioned that only two of these are applied to *dynamic* process systems.

Table 13: The expected diagnostic performance of the AIS with respect to other diagnostic systems. Adapted from Venkatasubramanian et al. 2002.

	Observer	**Digraphs**	**Abstraction Hierarchy**	**Expert Systems**	**QTA**	**PCA**	**Neural Nets**	**AIS**
Fast detection	Yes	No	No	Yes	Yes	Yes	Yes	**Yes**
Isolability	Yes	No	No	Yes	Yes	Yes	Yes	**No**
Robustness	Yes	Yes	Yes	Yes	Yes	Yes	Yes	**Yes**
Novelty Identifiability	No	Yes	Yes	No	No	Yes	Yes	**Yes**
Classification Error Est.	No	No	No	No	No	No	No	**Yes**
Adaptability	No	Yes	Yes	No	No	No	No	**Yes**
Explanation Facility	No	Yes	Yes	Yes	Yes	No	No	**No**
Modelling Requirement	N/A	Low	Low	Low	Low	Low	Low	**Low**
Storage and Computation	OK	N/A	N/A	OK	OK	OK	OK	**OK**
Multiple fault Identifiability	Yes	Yes	Yes	No	No	No	No	**Yes**

Chapter 3 discussed the methodology for process fault detection, based on negative selection. The procedure, as given in Chapter 3, commences by embedding data from a time series (as obtained from sensors) into a shape space. The two embedding parameters are the autocorrelation lag (L) and dimensionality determined by considering the eigenvalues as returned by PCA. Once the shape space is constructed and the self space is known, detectors may be generated using one of three methods. The system is hereafter ready to monitor the specific process for deviations from normal behaviour.

Three simulated case studies were considered in Chapters 4 through 6. In all cases, the hypercube vertex method illustrated superior performance. Refer to Figure 65 for a summary of these results.

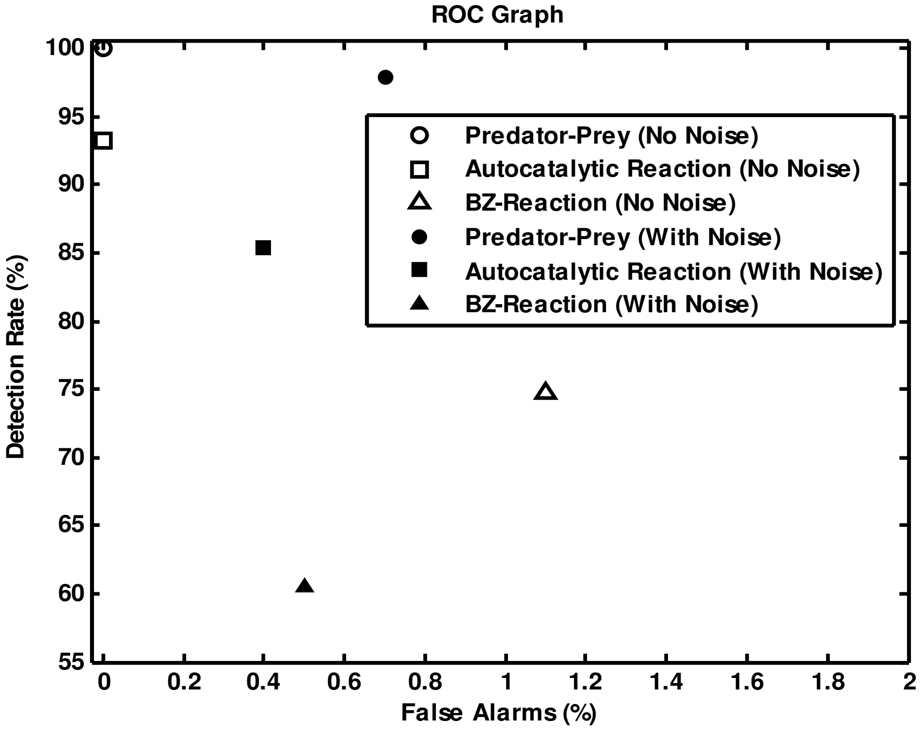


Figure 65: ROC graph representing the performance of the hypercube vertex algorithm in all three case studies, both with and without noise.

Although it is found that the hypercube vertex method performs the best in all three case studies, it should be noted that this is only true for the case studies considered and similar systems. The drawback of the hypercube vertex approach is its inability to upscale to large systems. By “large” systems one would refer to systems with higher dimensionalities. Consider a 10-dimensional system with 500 known self events. The maximum number of detectors that one would be able to generate in this case, would be $500\times2\times10=1E4$. Ten thousand detectors can easily "disappear” in a 10-dimensional shape space. It will also become increasingly difficult to cover the entire area around a point in higher dimensional hyperspace. Random generation of detectors might in this case be a more viable solution, since there is no limit to the number of detectors that can be generated. This topic might require further study though.

Finally, each method discussed throughout the thesis was investigated for its associated computational strain. It was found that hypercube vertex detectors require the most time per detector (27ms on average), while convex hulls require the least time (1.9ms on average). Random detectors require an average of 18ms per detector, for the case studies considered. The remainder of the process (space transformations, monitoring etc.) are fixed in their complexity and do not vary between case studies.

As a whole, it is therefore concluded that the field of AIS is promising in both academic and industrial applications.

Chapter 8: Recommendations

The final chapter in this thesis discusses several recommendations regarding the Application of AIS, future study and miscellaneous topics of possible concern.

Throughout the thesis, it is seen that various methods have their associated positive and negative attributes. Hypermutation for instance is effective in a well defined, preferably binary shape space, whereas random detectors are easily generated and perform reasonably well in most cases. It is therefore highly recommended that the nature of the shape space, the transformation between measure space and shape space and the method for generating detectors be taken in high regard when initiating an artificial immune system. Their pros and cons should be simultaneously considered as each of these factors influence the functioning of the others. When monitoring industrial processes, it is recommended that a real valued shape space be used, since the resolution of a binary system is simply insufficient for these processes. Further, it is recommended that the hypercube vertex method be used in real valued systems of which the dimensionality is not abnormally high. The number of detectors (N) and the detector affinity threshold (ε) play a decisive role in the detection capability of every system. Their values are thus to be optimised in order to achieve maximum reliability.

The future study of Artificial Immune Systems should perhaps include further investigation into real processes and practical implications of using AIS in monitoring such systems. Work may be done to provide AIS with an explanation facility. Isolability is another shortcoming of AIS; future study might involve finding a way in which artificial immune systems can isolate errors and provide a possible solution. This would combine the detection abilities of AIS with a control system to ensure flawless operation. The hypermutation algorithm is mentioned but not studied in this thesis. Some experiments were however done which suggest that this method might provide a viable alternative to the methods mentioned in this thesis, provided of coarse that at least some knowledge of the nonself space is known at the time of detector generation. Further detectors may then be mutated during monitoring of the process, which allows for exceptional adaptability.

Further research should be done on the impact of chaotic systems on AIS. A short study on this matter during the course of this project revealed that chaos might pose a problem during monitoring. The self space of chaotic systems includes the presence of holes that may be large enough to disrupt the notion of a single continuous self space. Spiralling self spaces may require detectors with varying affinities. Multiple detector sets may be required in cases like these. This leads to the study of hybrid methods. Various different types of detectors may be combined to construct hybrid Artificial Immune Systems.

Chapter 9: References

Alciaturi C.E., Escobar M.E., Est´eves I. 2005. The use of the autocorrelation function in modelling of multivariate data. Analytica Chimica Acta 553 pp 134-140.

Anchor K.P., Zydallis J.B., Gunsch G.H. and Lamont G.B. 2002. Extending the Computer Defense Immune System: Network Intrusion Detection with a Multiobjective Evolutionary Programming Approach. In Proceedings of the First International Conference on Artificial Immune Systems (ICARIS-2002).

Anderson, T. W. 1984. An introduction to multivariate statistical analysis. New York: Wiley.

Bradley D.W. and Tyrell A.M. 2002. A Hardware Immune System for Benchmark State Machine Error Detection. In Congress on Evolutionary Computation. Part of the World Congress on Computational Intelligence, pp 813-818.

Bureau of Labor Statistics, 1998. Occupational injuries and illnesses in the united states by industry. Washington, DC: Government Printing Office.

Carpenter C.A., Grossberg S. 1988 The Art of Adaptive Pattern Recognition by a Self-Organizing Neural Network. Computer 21 (3) pp 77-88.

Cheung J. T. and Stephanopoulos G. 1990. Representation of process trends part I. A formal representation framework. Computers and Chemical Engineering 14 (4-5), pp 495-510.

Cybenko G. 1989. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems 2, pp 303-314.

Dasgupta D. 1999. Artificial Immune Systems and Their Applications. Springer-Verlag Berlin Heidelberg.

Dasgupta, D., Forrest, S., 1995. Tool breakage detection in milling operations using negative selection algorithm. Technical Report No CS95-5, Department of Computer Science, University of New Mexico, Albuquerque, NM, USA.

Dasgupta, D., Forrest, S., 1999. An anomaly detection algorithm inspired by the immune system. Artificial Immune Systems and Their Applications. Springer-Verlag, Inc., pp 262-277 (chapter 14).

Deaton R., Garzon M. Rose J.A., Franceschetti, D.R., Murphy, R.C., and Jr. S.E.S. 1997. DNA Based Artificial Immune System for Self-Nonself Discrimination. In Proc. IEEE Int. Conf. Systems, Man, and Cybernetics. IEEE.

De Casto L.N., Von Zuben F.J. 2000. Artificial Immune Systems: Part II - a Survey Of Applications. Technical Report DCA-RT 02/00. Computer Science and Computer Engineering. Pacific Lutheran University.

De Casto L.N., Von Zuben F.J. 2002. Learning and Optimization Using the Clonal Selection Principle. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 3, pp 239-251.

Dong A., Sun Z., Jia H. 2004. A Cosine Similarity-Based Negative Selection Algorithm for Time Series Novelty Detection. Mechanical Systems and Signal Processing 20 (2006) pp 1461-1472.

Eli, B., Richard, C., Geoffrey, S., 2000. Immunology, fourth edition. A John Wiley & Sons, Inc., USA.

Engin O., Döyen A., 2004. A New Approach to Solve Hybrid Flow Shop Scheduling Problems by Artificial Immune System. Future Generation Computer Systems 20 (2004) pp 1083-1095.

Esponda F., Forrest S., Helman P. 2004. A Formal Framework for Positive and Negative Detection Schemes. IEEE, Transactions on Systems, Man, and Cybernetics. Part B Vol. 34, pp 357-373.

Farmer, J., Packard, N., Perelson, A. 1986. The immune system, adaptation and machine learning. Physica D, 22 pp 187-204.

Forrest, S., Allen, L., Perelson, A.S., Cherukuri, R. 1994. Self-nonself discrimination in a computer. In: Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy, Los Alamitos, CA, USA.

Forrest S. and Hofmeyr S. 2000. Immunology as Information Processing. In Segel, L. and Cohen, I., editors, Design Principles for Immune System and Other Distributed Autonomous Systems. Oxford University Press.

Fukunaga K. 1972. Introduction to statistical pattern recognition. New York: Academic press.

Fyfe C., Jain L. 2004. Teams of Intelligent Agents Which Learn Using Artificial Immune Systems. Journal of Network and Computer Applications 29 (2006) pp 147-159

Garrett, S.M. 2005. How do we evaluate artificial immune systems? Evolutionary Computation 13 (2), pp 145-178.

Gertler J. 1989. Intelligent supervisory control. In A. E. Nisenfeld & J. R. Davis (Eds.), Artificial intelligence handbook, Volume II. Research Triangle Park, NC: USA.

Hofmeyr S. and Forrest S. 1999. Immunity by design: An Artificial Immune System. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99), pp 1289–1296.

Hofmeyr S. and Forrest S. 2000. Architecture for the Artificial Immune System. Evolutionary Computation, 8(4): pp 443-473.

Holyoak M., Lawler S.P. 1996. Persistence of an Extinction-Prone Predator-Prey Interaction Through Metapopulation Dynamics. Ecology: Vol. 77, No. 6, pp1867-1879

Hotteling H. 1947. Multivariate Quality Control Illustrated by the Testing of Sample Bombsight. Selected Techniques of Statistical Analysis, McGraw Hill. pp113-184.

Himmelblau D.M. 1978. Fault Detection and Diagnosis in Chemical and Petrochemical Processes. Amsterdam: Elsevier Press.

Hunt J.E. Cooke D.E. 1996. Learning Using an Artificial Immune System. Journal of Network and Computer Applications (1996) 19, pp 189-212.

Jackson J. E. 1991. A user's guide to principal components. New York: Wiley-Interscience.

King R.L., Russ S.H., Lambert A.B., Reese D.S. 2001. An Artificial Immune System for Intelligent Agents. Future Generation Computer Systems 17 (2001) pp 335-343.

Lee, J.S., and Chang, K.S. 1996. Applications of chaos and fractals in process systems engineering. Journal of Process Control, Vol. 6, No. 2/3, pp 71-87.

Lees F.P. 1996. Loss prevention in process industries: hazard identification, assessment and control. London: Butterworth-Heinemann.

Lu P., Teulilo A. 2002. Oscillating Chemical Reaction. PHYS3301 Scientific Computing pp 1-15.

Luh G.C., Cheng W.C. 2005. Immune Model-Based Fault Diagnosis. Mathematics and Computers in Simulation, Vol 67, Issue 6, pp 515-539.

Marr D. and Hildreth E. 1980. Theory of edge detection. Proceedings of Royal Society London 207, pp 187-217.

McGraw-Hill Economics. 1985. Survey of investment in employee safety and health. NY: McGraw-Hill Publishing Co.

National Safety Council. 1999. Injury facts 1999 Edition, Chicago: National Safety

Council.

Nimmo, I. 1995. Adequately address abnormal situation operations. Chemical Engineering Progress 91 (9), pp 36-45.

Pearson K. 1901. On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine Series B 2, pp 559-572.

Perelson A.S., Oster G.F. 1979. Theoretical studies of clonal selection: Minimal antibody repertoire size and reliability of self-non-self discrimination, Journal of Theoritical Biology 81 (4) pp 645-670.

Stibor T., Mohr P., Timmis J. 2005. Is Negative Selection Appropriate for Anomaly Detection? Gecco'05, pp 321-328.

Tarakanov A., Dasgupta D. 2001. A Formal Model of An Artificial Immune System. BioSystems 55 (2000) pp 151-158.

Timmis, J., Neal, M., Hunt, J. 2000. An artificial immune system for data analysis. Biosystems 55, pp 143-150.

Venkatasubramanian V., McAvoy T.J. 1992. Special Issue on Neural Network Applications in Chemical Enginering. Computers and Chemical Engineering 16(4). R5-R6.

Venkatasubramanian V., Rengaswamy R., Yin K., Kavuri S.R. 2002a. A Review of Process Fault Detection and Diagnosis. Part I: Quantitative Model-Based Methods. Computers and Chemical Engineering 27 pp 293-311.

Venkatasubramanian V., Rengaswamy R., Yin K., Kavuri S.R. 2002b. A Review of Process Fault Detection and Diagnosis. Part III: Process History Based Methods. Computers and Chemical Engineering 27 pp 327-346.

Weisstein E.W. 2003. Lotka-Volterra Equations. From MathWorld-A Wolfram Web Resource. http://mathworld.wolfram.com/Lotka-VolterraEquations.html.

Wierzchon, S.T. 1999. Generating antibody strings in an artificial immune system. Technical Report ICS PAS Report No. 892, Institute of Computer Science, Polish Academy of Sciences.

Wierzchon, S.T. 2000. Discriminative power of the receptors activated by k-contiguous bits rule. Journal of Computer Science and Technology, 1(3): pp 1-13.

Wierzchon, S.T. 2002. Function optimization by the immune metaphor. Task Quarterly, 6(3): pp 1-16.

Williamson M.M. 2002. Biologically-Inspired Approaches to Computer Security.

Technical Report HPL-2002-131, HP Labs, Bristol, UK. Available from http://www.hpl.hp.com/techreports/2002/HPL-2002-131.html.

Willsky A.S. 1976. A survey of design methods for failure detection in dynamic systems. Automatica 12, pp 601-611.

Wold S. 1978. Cross-validatory estimation of the number of components in factor and principal components models. Technometrics 20 (4), pp 397-405.

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. Chemometrics and Intelligent Laboratory Systems 2 (1-3), pp 37-52.

Yang X., Aldrich C., Maree C. 2006. Detecting Change in Dynamic Process Systems with Immunocomputing. Minerals Engineering, article in press.

Zheng H., Zjang J. and Nahavandi S., 2004. Learning to Detect Texture Objects by Artificial Immune Approaches. Future Generation Computer Systems 20 (2004) pp 1197-1208.

APPENDIX

DIAGNOSTIC MONITORING OF DYNAMIC PROCESS SYSTEMS USING ARTIFICIAL IMMUNE SYSTEMS

By: Charl Maree
Supervised by: Prof. Chris Aldrich

Department of Process Engineering,
University of Stellenbosch, South Africa

13 October 2006

PROCESS ENGINEERING
STELLENBOSCH·UNIVERSITY

Outline of Presentation

- Introduction
- The Natural Immune System
- The Artificial Immune System
    - Artificial Immune Algorithms
    - The Negative Selection Algorithm
- Methodology
    - Embedding the Self Space
    - Choice of Matching Rule
    - Generating Detectors
- Case Studies
    - Lokta-Volterra Process
    - Autocatalytic Reaction
    - Belousov-Zhabotinsky Reaction
- Conclusion
- Recommendations
- Acknowledgements
- Questions

Introduction

- Process anomaly detection is vital on modern process plants.
  - Approx. 70% of industrial accidents are caused by human errors.
  - Petrochemical industries in the USA alone lose an estimated 20 billion dollars annually.
  - Two of the most devastating chemical plant accidents, namely Union Carbide’s Bhopal (India) and Occidental Petroleum's Piper Alpha *might* have been prevented.
  - The Chernobyl disaster affected 3.7 million people across Europe.

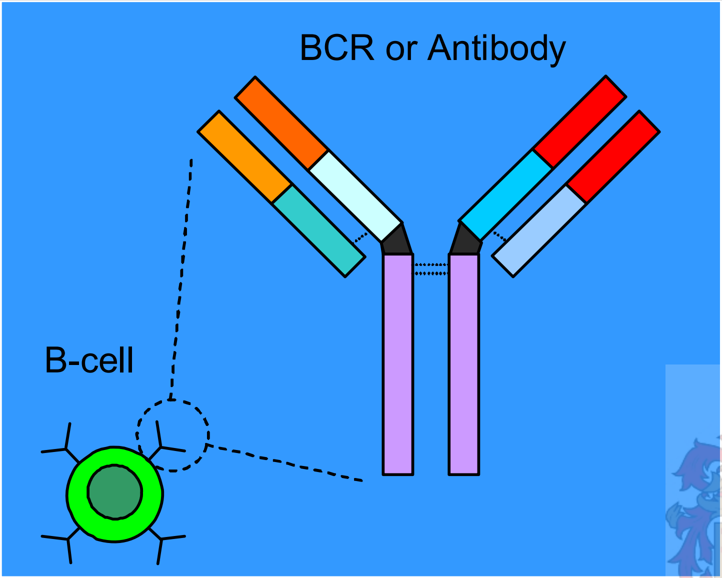
Introduction (cont.)

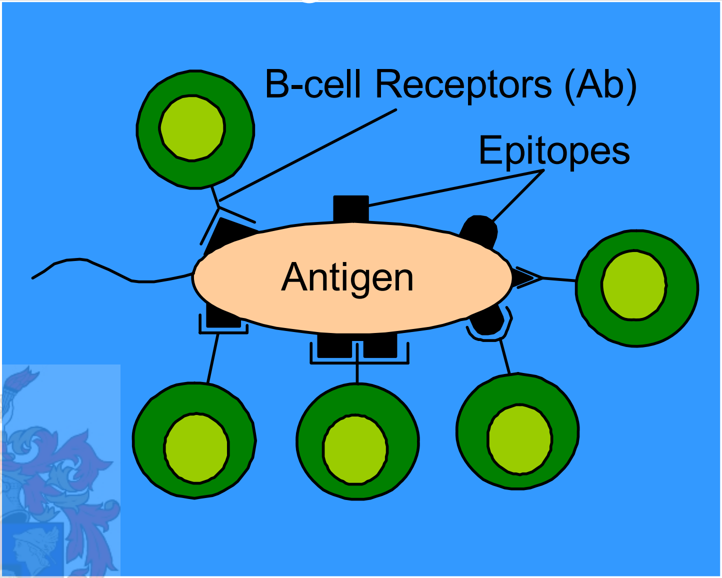
- Various established statistical methods for anomaly detection exist
- Unfortunately they are only defined when process operation meets certain assumptions, such as:
  - normal distributions of variables
  - linear correlations between variables
  - no auto- or cross-correlation in variables
- Immunocomputing is a promising analytical paradigm that has emerged very recently.

Introduction (cont.)

- Definition: A computational system based on the natural immune system, used for monitoring or diagnosis of process data.
- Why model the *immune system*?
  - Robustness
  - Adaptability
  - Multiple fault identifiability
  - Memory / Learning

Natural Immune Systems





- Recognition is based on the *complementarity* between *receptors* and *epitopes*.
- This analogy is applicable in a large variety of contexts:
  - Computer security systems
  - Pattern recognition
  - Optimisation
  - Detection of anomalies
  - **Process engineering!**

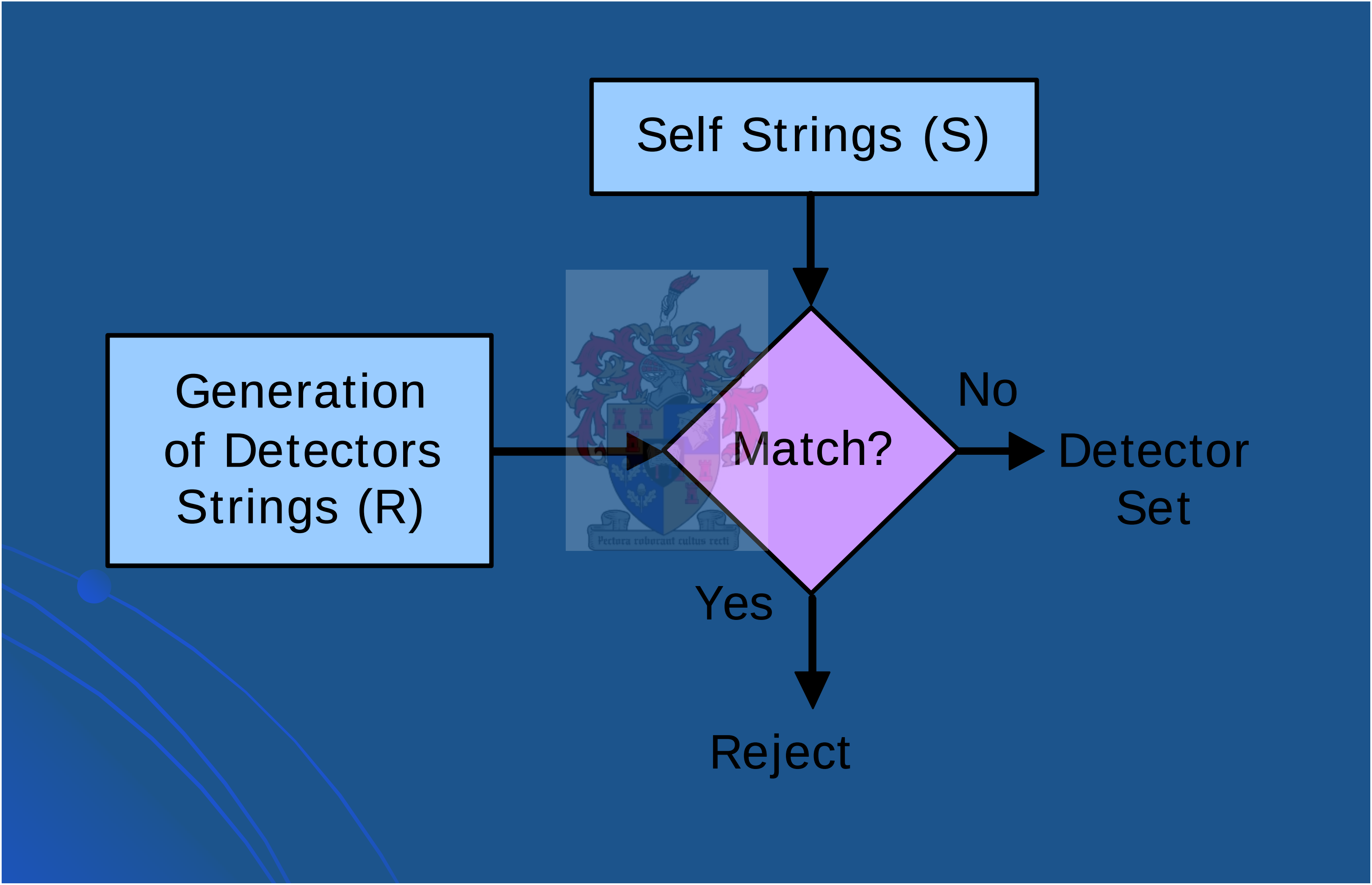
Artificial Immune Systems

- Do not attempt to *perfectly* match the natural immune system.
- Lymphocytes/Phagocytes → Detectors
- Antigens → Process anomalies
- Known Proteins → Historical process data (Self)
- Immune Response → Detector alarm

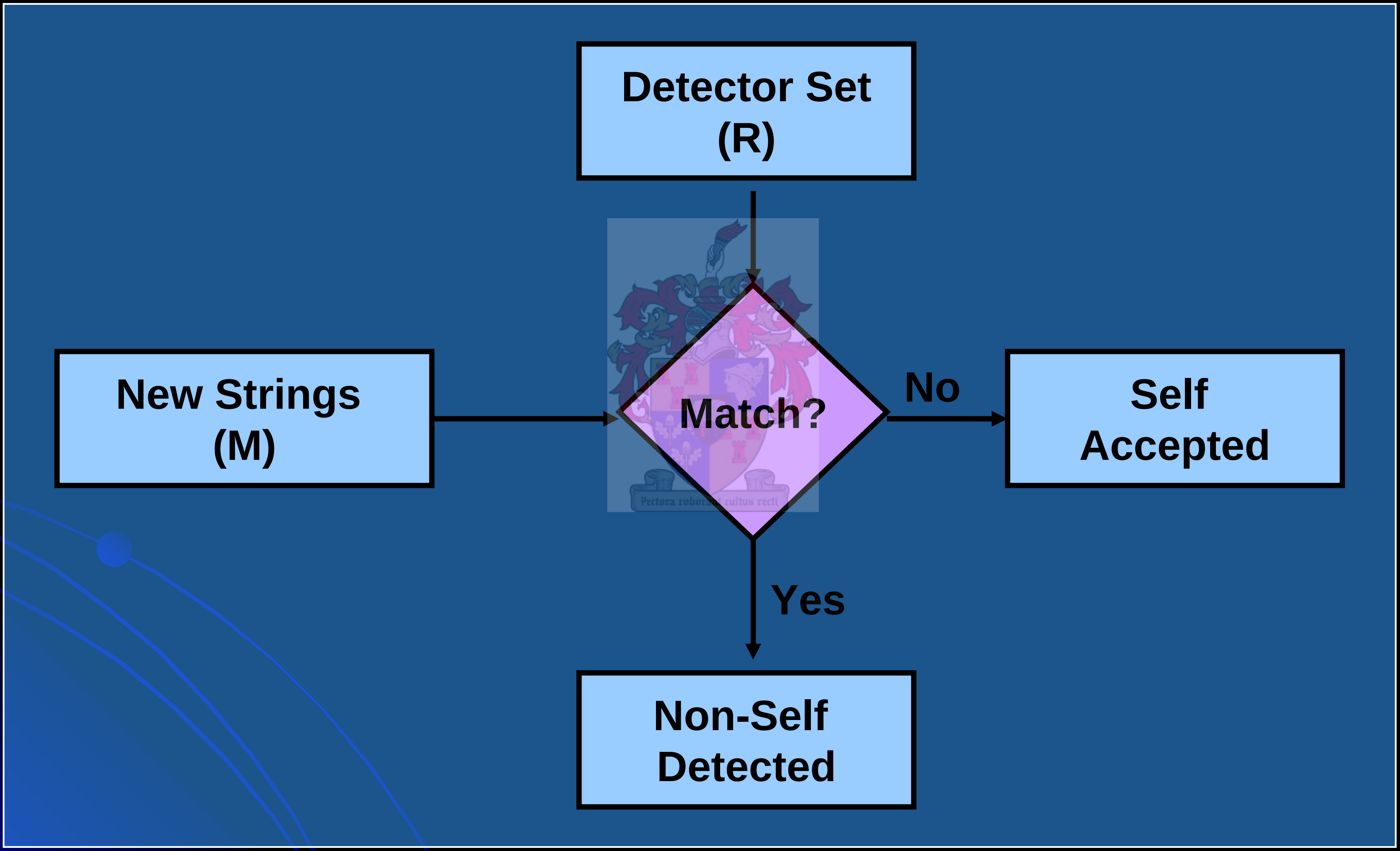
Artificial Immune Algorithms

- Sub-Classes in an AIS:
  - Positive and Negative Selection
  - Clonal Selection and Hypermutation
  - Artificial Immune Networks
  - Danger Theory

Negative Selection Algorithm: Censoring



Negative Selection Algorithm: Monitoring



Embedding the Self Space

- Retrieve time series data
  - Real plant data
  - Simulated process
- Calculate the autocorrelation function R_L of the time series.
  - Choose the smallest L that satisfies $R_L = 0$

$$R_L(t) \equiv \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} f(L) f(t+L) dL$$

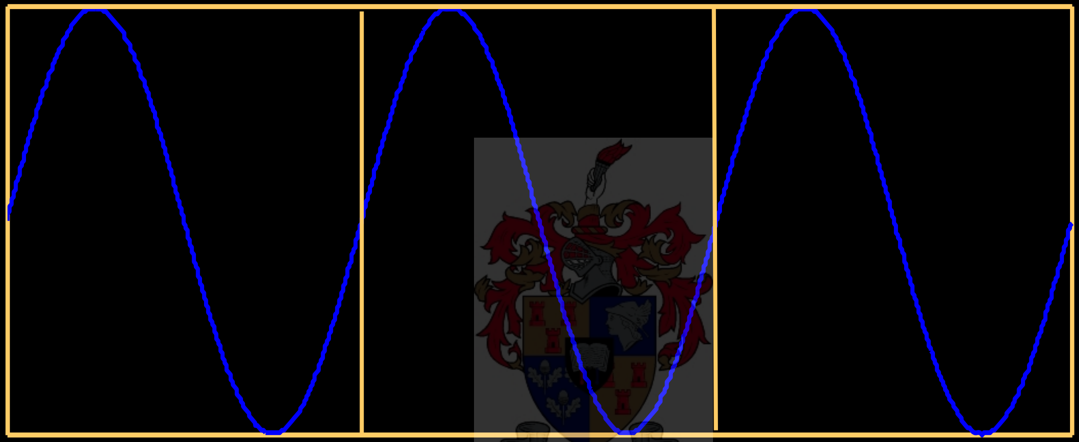
$$R_L \to -1 \qquad \textit{Alternating Series}$$

$$R_L \to \ \ 0 \qquad \textit{Random Series}$$

$$R_L \to +1 \qquad \textit{Smooth Series}$$

Embedding the Self Space (Cont.)

- “Split” time series into windows of length L.



- Stack windows to form matrix.
- For more than one time series, i.e. Temperature, Pressure and Concentration; multiple matrices are combined by joining rows

Embedding the Self Space (Cont.)

- Embedding is defined by the following mathematical equation

$$S_{n,ij} = T_{n,k}$$

$$where:$$

$$k = (i-1) \times L + j$$

$$n \quad is \quad the \quad number \quad of \quad time \quad series$$

$$S = \begin{bmatrix} S_{1,IJ} & S_{2,IJ} & \cdots & S_{N,IJ} \end{bmatrix}$$

$$where$$

$$I = 1,2,.., L$$

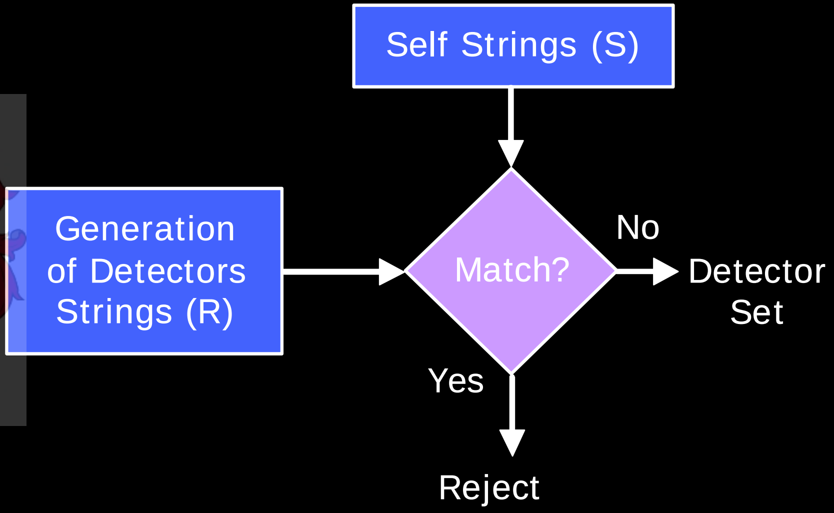
$$J = 1,2,.., \min(j_n)$$

Embedding the Self Space (Cont.)

- Apply PCA to determine the dimensionality of the self space.
- Approx. 90% of the original data variance should be captured.
- A smaller matrix (in dimensionality) is obtained which describes the self space.
- Embedding parameters:
  - Autocorrelation Lag (L)
  - Cumulative sum of PCA eigenvalues.

Choice of Matching Rule

- r-Contiguous bits (RCB)
- Euclidean Distance (ED)
- Other
  - Classic Hamming distance
  - Manhattan distance (aka Rectilinear)
  - Vector Cosine measure
  - L-norm distance
- Key Importance:
  - Ability to describe distance in the event space accurately



Self Strings (S)

Generation of Detectors Strings (R)

Match?

No

Detector Set

Yes

Reject

r-Contiguous Bits Matching Rule

- Best suited for a binary alphabet
- Example:
  - Let Ab = [1 1 0 1 1 0 0]
  - Let Ag = [1 0 0 1 1 0 1]
  - match(Ab,Ag) = true for r <= 4

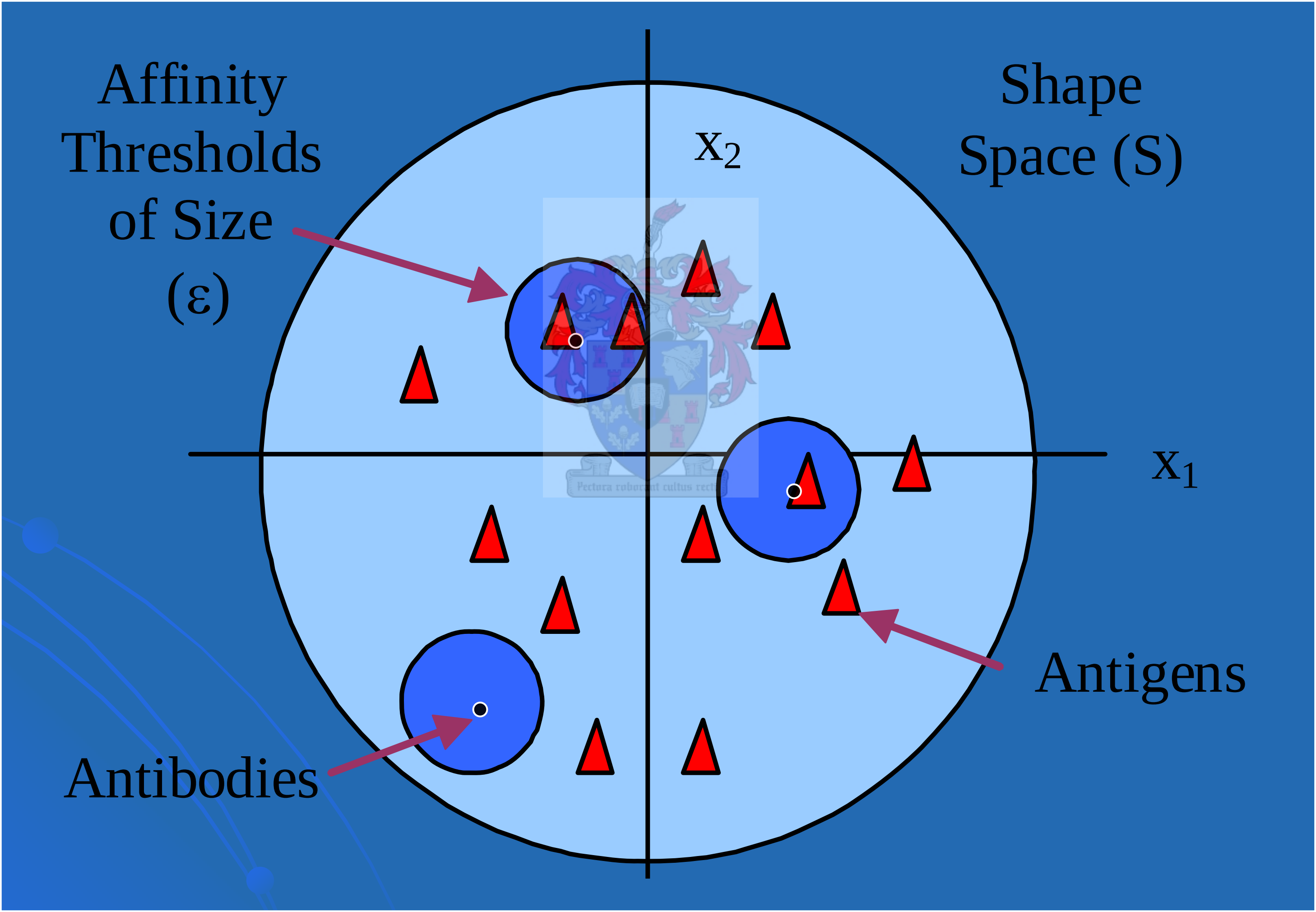
Euclidean Distance Matching Rule

- Suited for a larger alphabets, i.e. real data
- Affinity is related to distance:

$$D = \sqrt{\sum_{i=1}^{L} (Ab_i - Ag_i)^2}$$

- match(Ab,Ag) = true if D < threshold

Affinity Threshold in the Ab-Ag Shape Space



Affinity Thresholds of Size (ε)

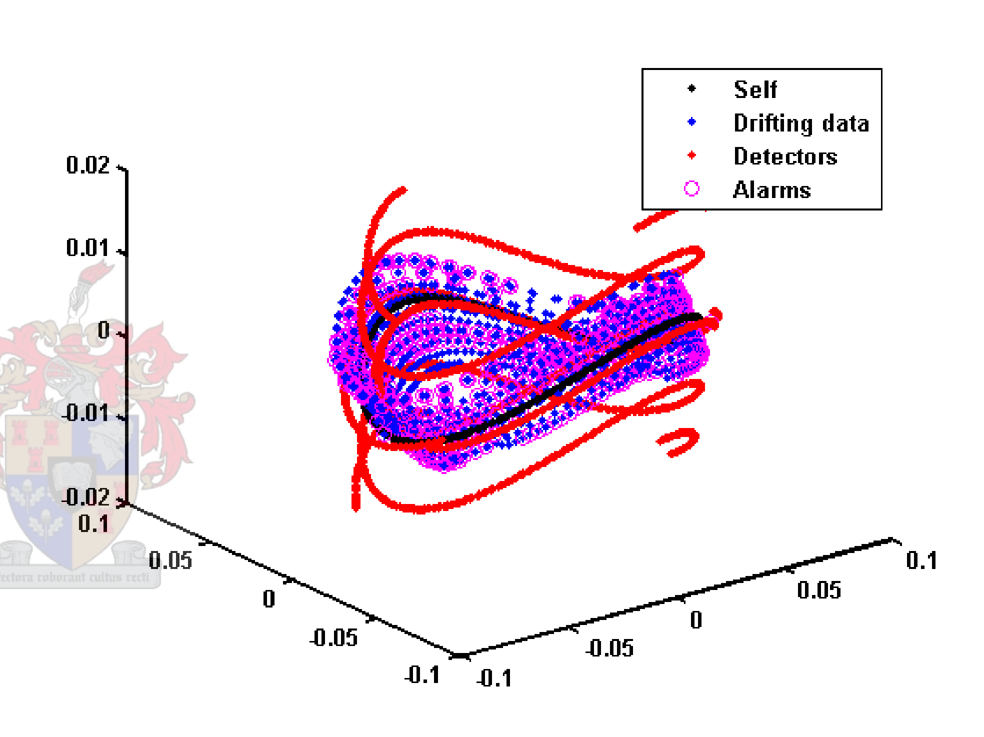
Shape Space (S)

Antigens

Antibodies

Generating Detectors

- Random
- Convex Hulls
- Hypercube Vertices
- Important factors:
  - Time to generate *x* detectors
  - Ability of detectors to identify process anomalies
  - False alarms generated



Random Detectors

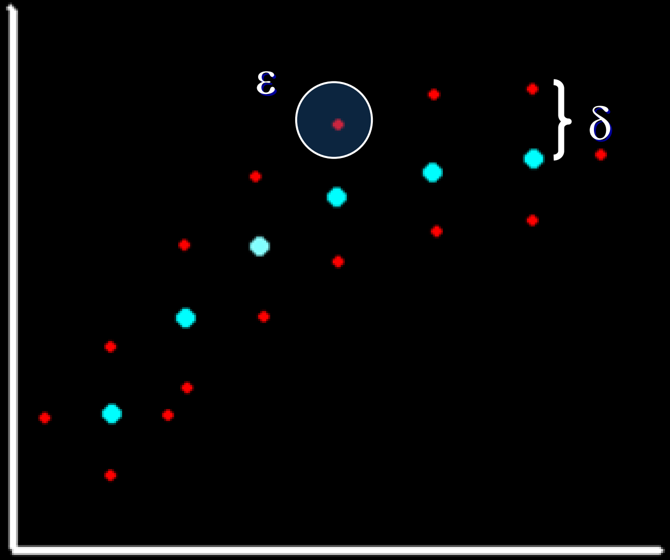
- Generate a random string containing coordinates in the n-Dimensional shape space.
- Test for a match with any of the points in the self space.
- If match = false, add the string to the detector pool.

Convex Hulls

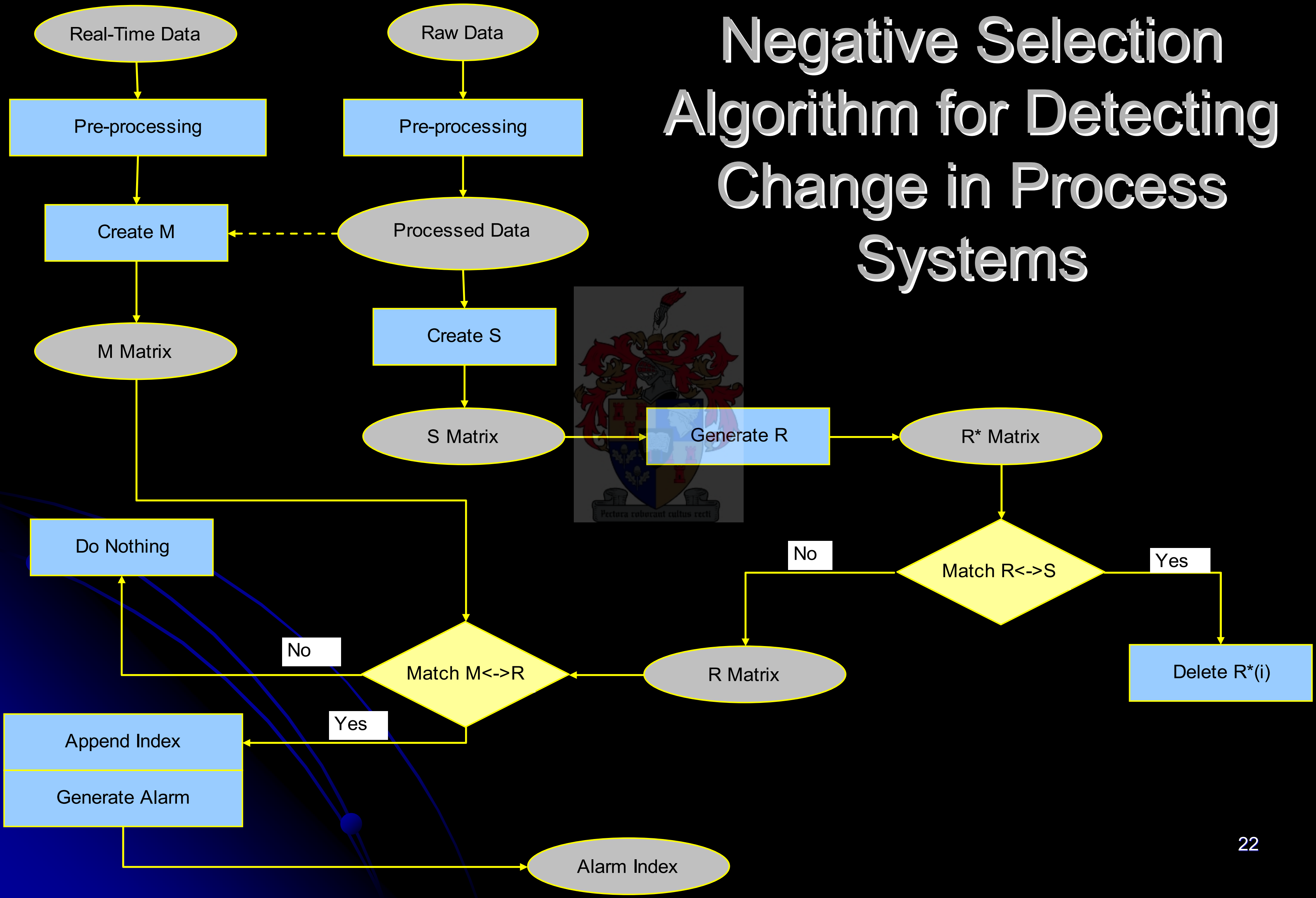
- Identify the boundary of the self space
  - Add Gaussian noise of size epsilon (ε)
  - Fit the convex hulls around the self space
  - Remove a layers one at a time until inner boundaries are defined
- Position detectors on both outer and inner boundaries

Hypercube Vertices Method

- Position detectors on vertices of a hypercube around each self event
- Remove unwanted detectors
- Two Parameters:
  - Epsilon (ε)
    - Specifies sensitivity
    - Sensor radius around detector
  - Delta (δ)
    - Specifies flexibility
    - Minimum distance to nearest self string



Negative Selection Algorithm for Detecting Change in Process Systems



Real-Time Data

Pre-processing

Create M

M Matrix

Raw Data

Pre-processing

Processed Data

Create S

S Matrix

Generate R

R* Matrix

Match R<->S

No

Yes

R Matrix

Delete R*(i)

Match M<->R

No

Do Nothing

Yes

Append Index

Generate Alarm

Alarm Index

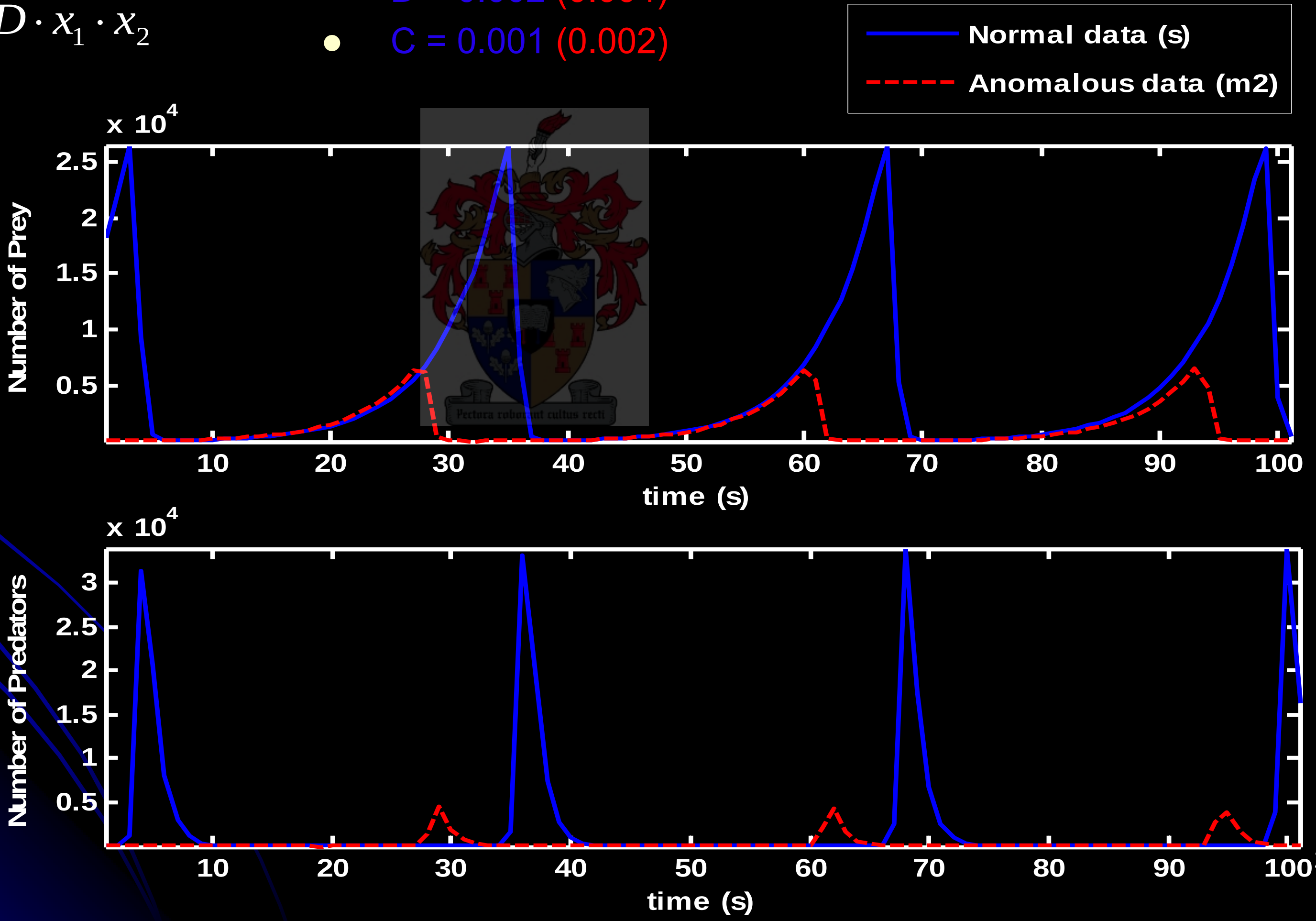
Simulation 1: Lokta-Volterra Process

$$\frac{dx_1}{dt} = k_1 \cdot x_1 - C \cdot x_1 \cdot x_2$$

$$\frac{dx_2}{dt} = -k_2 \cdot x_2 + D \cdot x_1 \cdot x_2$$

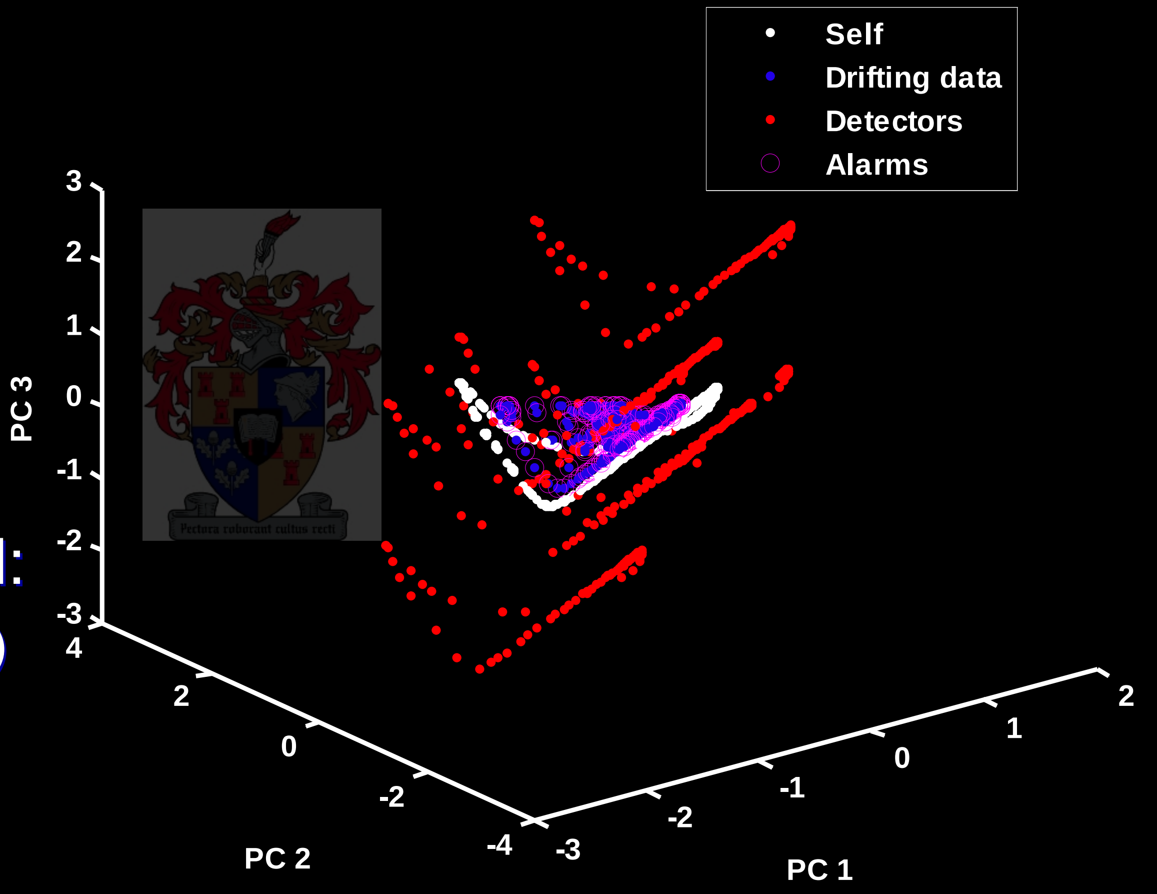
where:

- $k_1 = 2$
- $k_2 = 10$
- D = 0.002 (0.004)
- C = 0.001 (0.002)

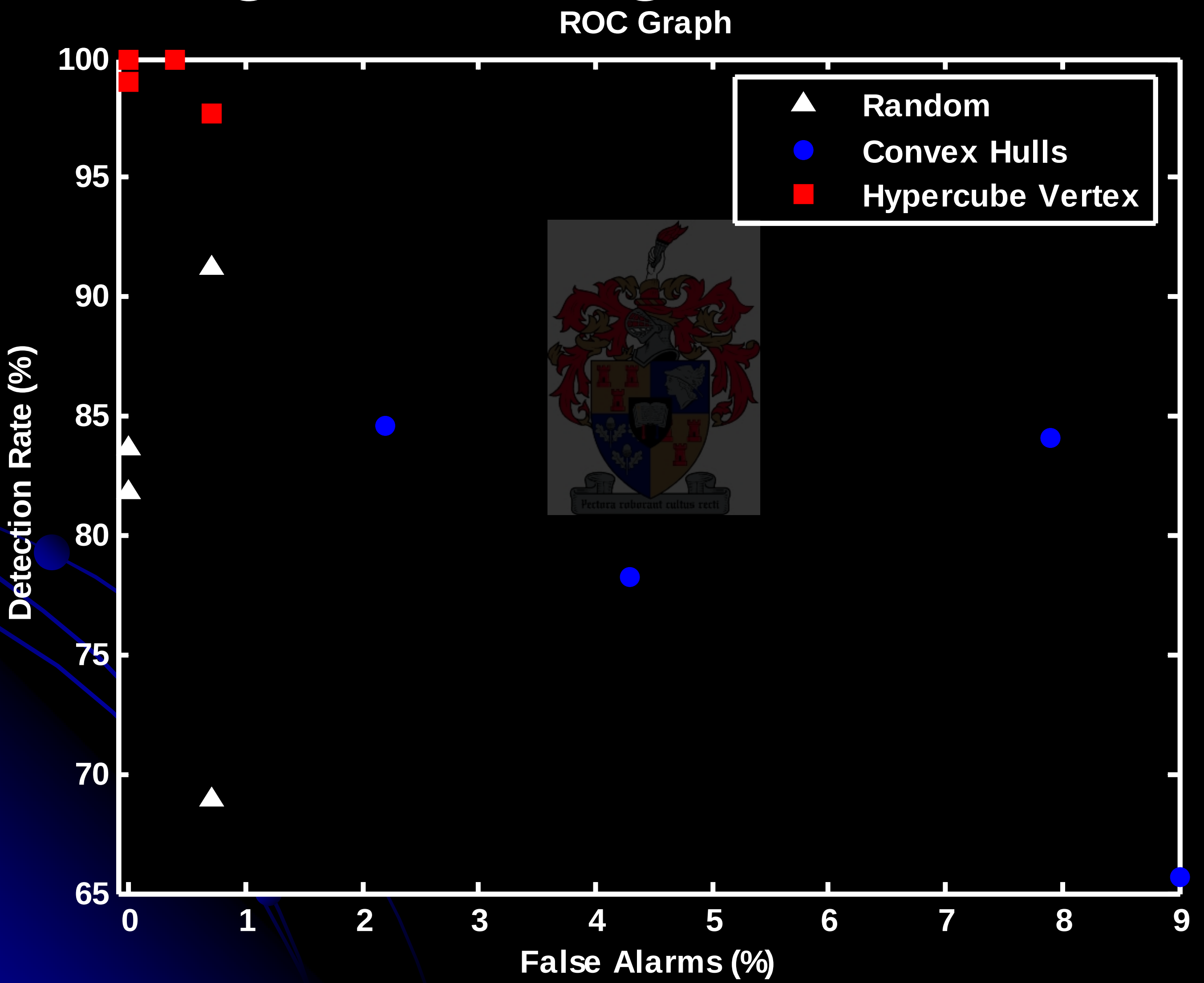


Self Space and Detection of Process Anomalies

- S → Attractor
- R → Detectors
- M → Data to be monitored
- Alarms marked with circles
- Anomalies Identified:
  - Drifting: 100% (98%)
  - Drifted: 100% (96%)
  - Error: 0% (0.7%)



Summary of different methods for generating detectors



Simulation 2: Autocatalytic Process

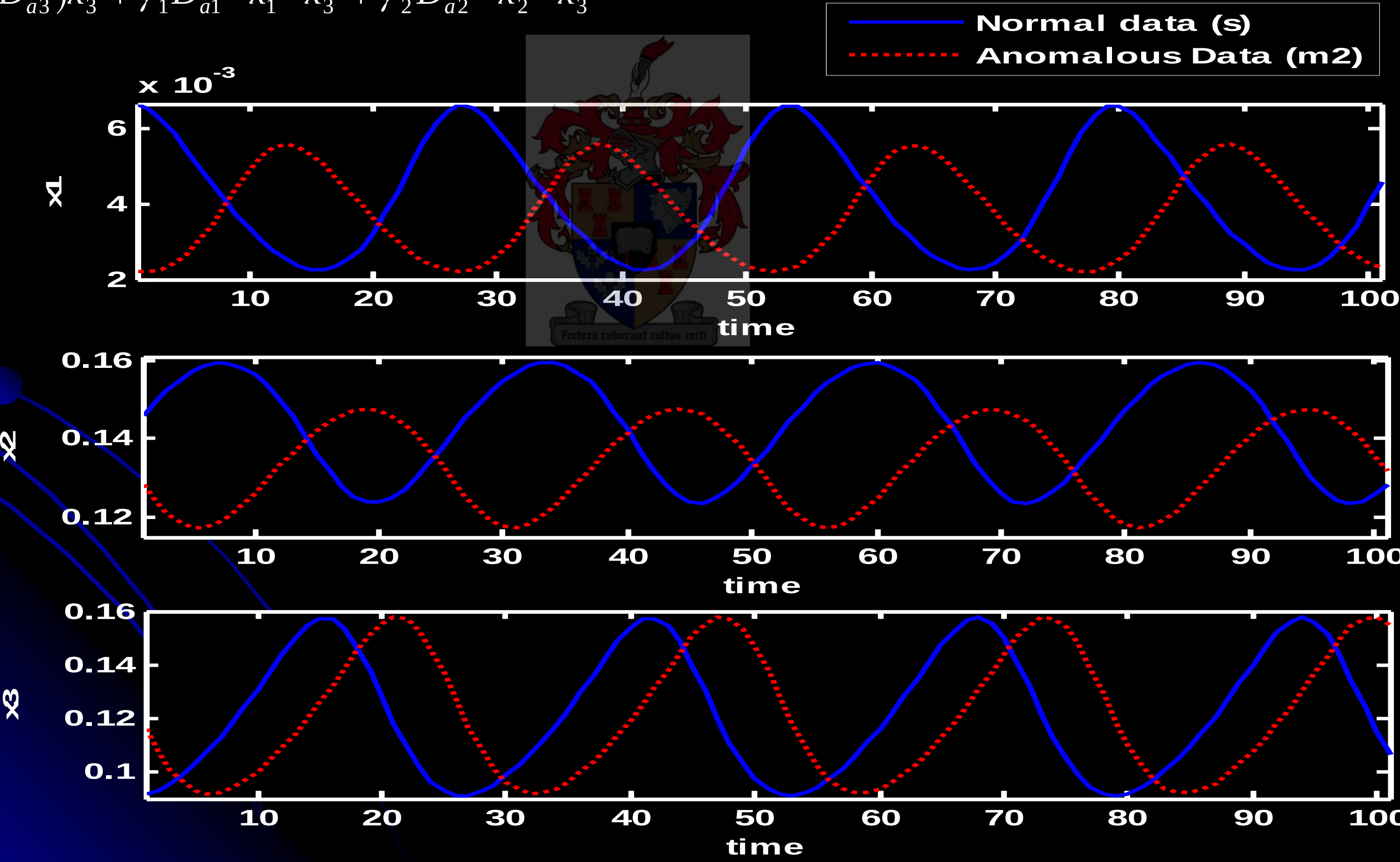
$$\frac{dx_1}{d\tau} = 1 - x_1 - D_{a1} \cdot x_1 \cdot x_3^2$$

$$\frac{dx_2}{d\tau} = 1 - x_2 - D_{a2} \cdot x_2 \cdot x_3^2$$

$$\frac{dx_3}{d\tau} = 1 - (1 + D_{a3})x_3 + \gamma_1 D_{a1} \cdot x_1 \cdot x_3^2 + \gamma_2 D_{a2} \cdot x_2 \cdot x_3$$

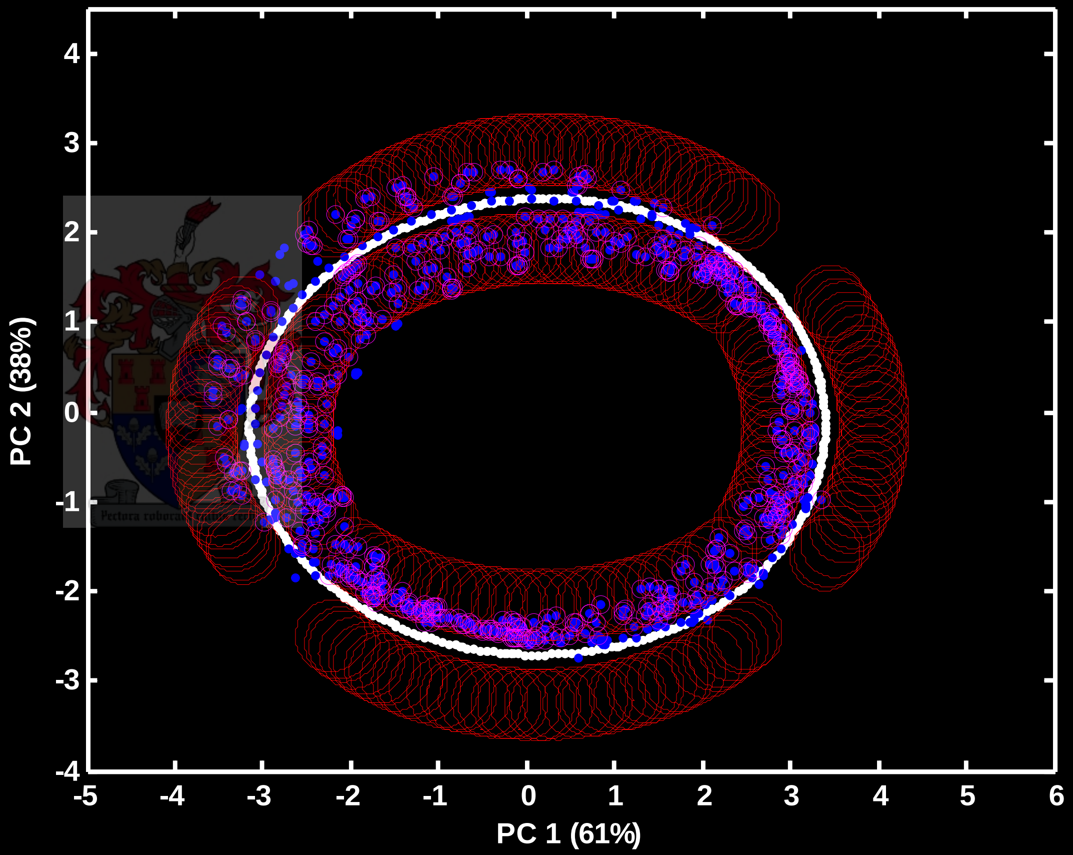
$D_{a1} = 18000$ $D_{a2} = 400$ $D_{a3} = 80$

$\gamma_1 = 3.5$ (3.7) $\gamma_2 = 6.2$ (6.4)

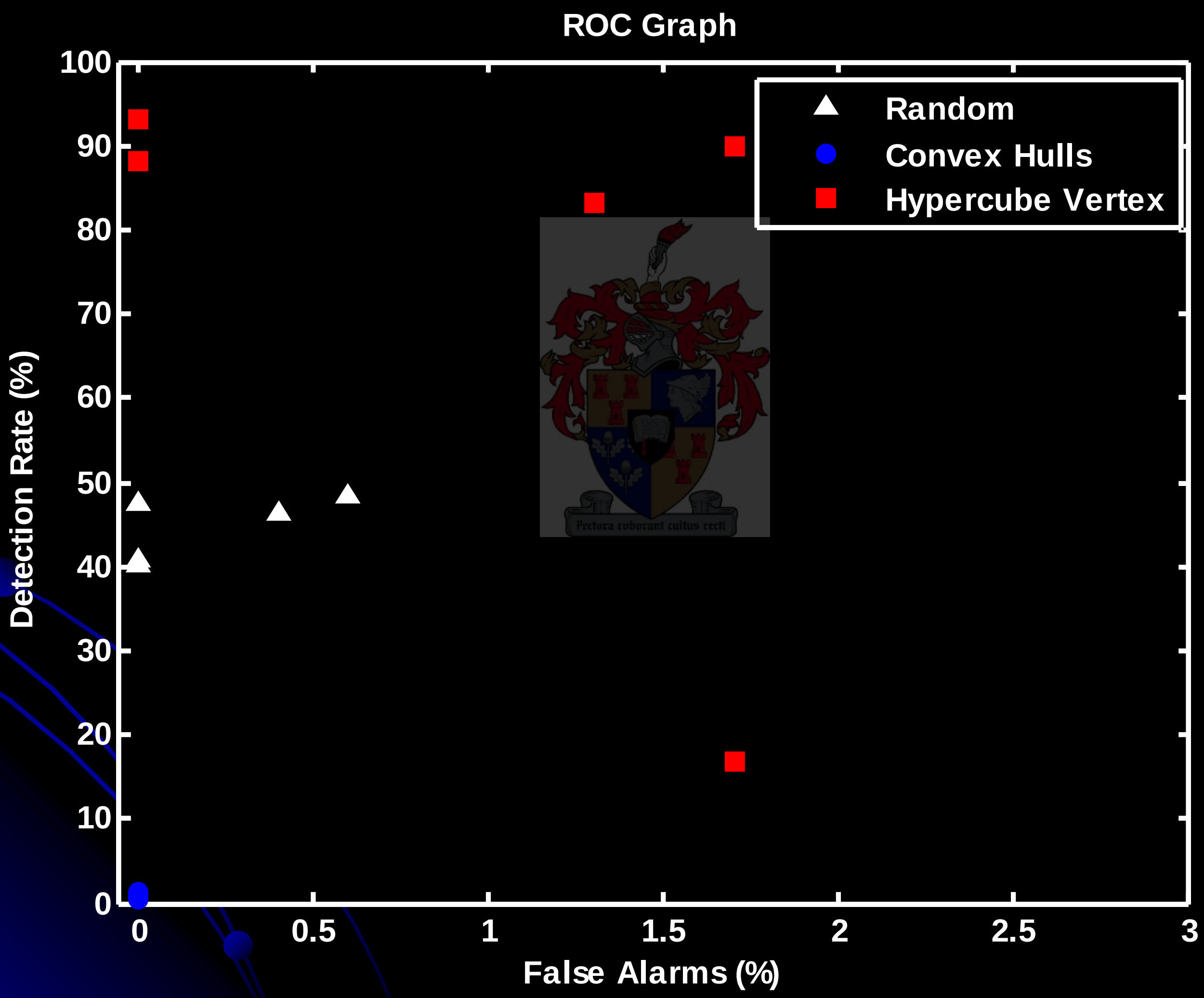


Self Space and Detection of Process Anomalies

- S → Attractor
- R → Detectors
- M → Data to be monitored
- Alarms marked with circles
- Anomalies Identified:
  - Drifting: 100% (98%)
  - Drifted: 80% (72%)
  - Error: 1.7% (0.4%)



Summary of different methods for generating detectors



Simulation 3: Belousov-Zhabotinsky Reaction

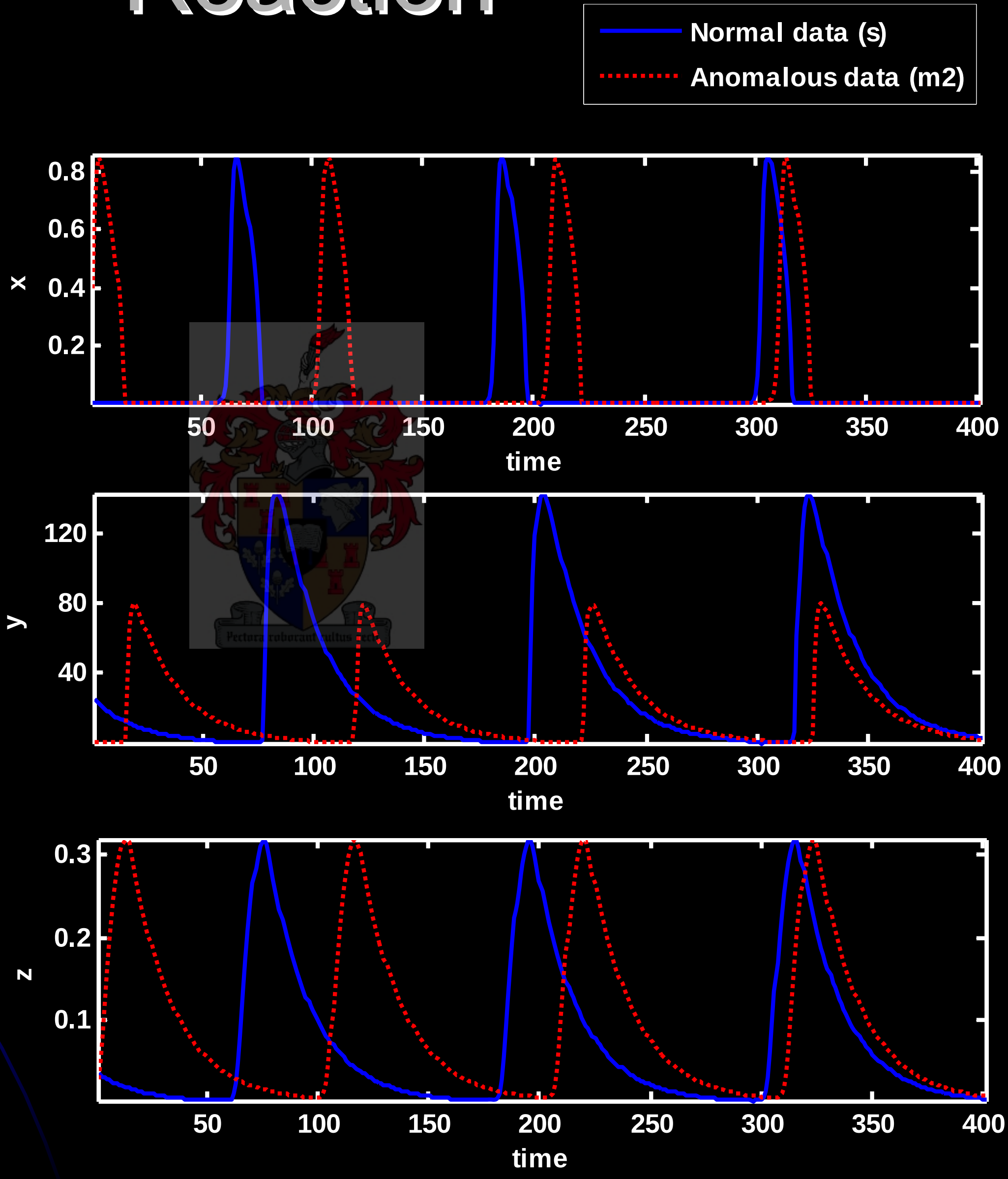
$$\frac{dx}{d\tau} = \frac{1}{c_1}\left[qx - xy + x(1-x)\right]$$

$$\frac{dy}{d\tau} = \frac{1}{c_2}\left[-qy - xy + fz\right]$$

$$\frac{dz}{d\tau} = x - z$$

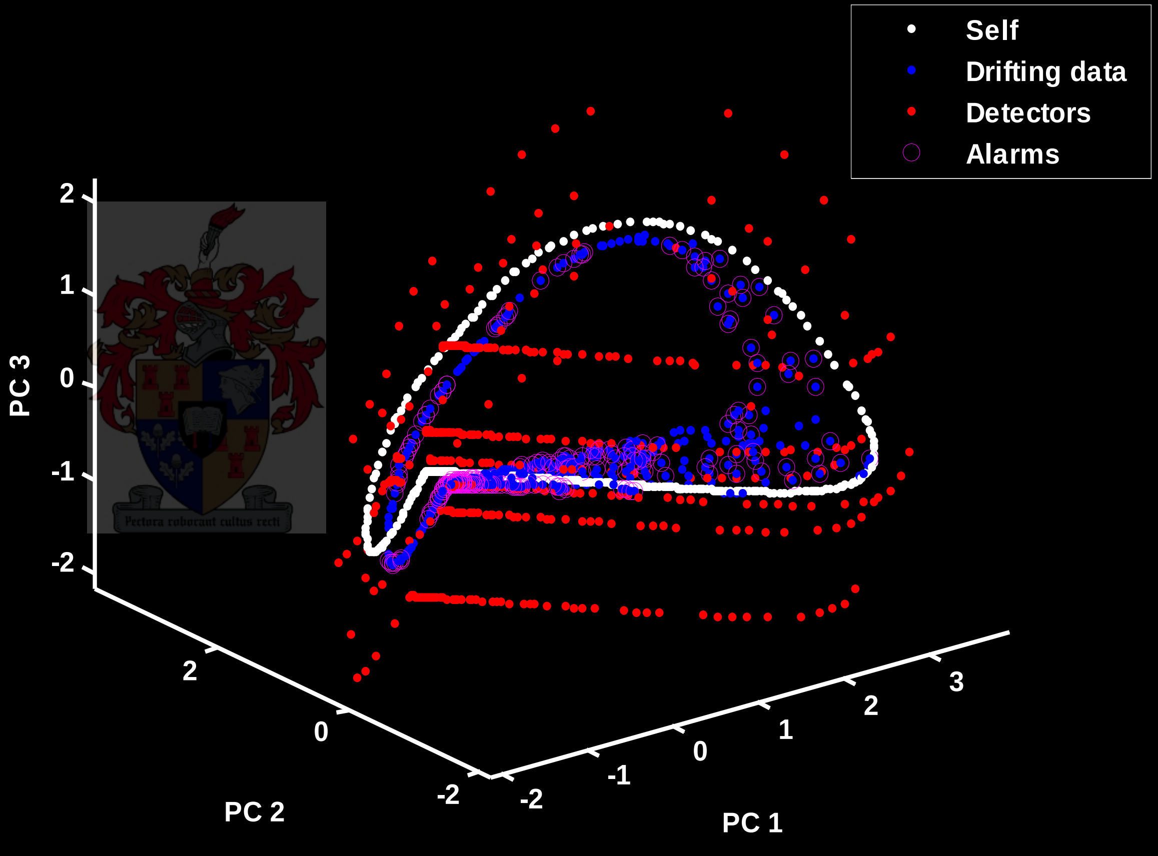
Where:

- $c_1 = 0.04$
- $c_2 = 0.0002$
- $q = 0.0008$ (0.0016)

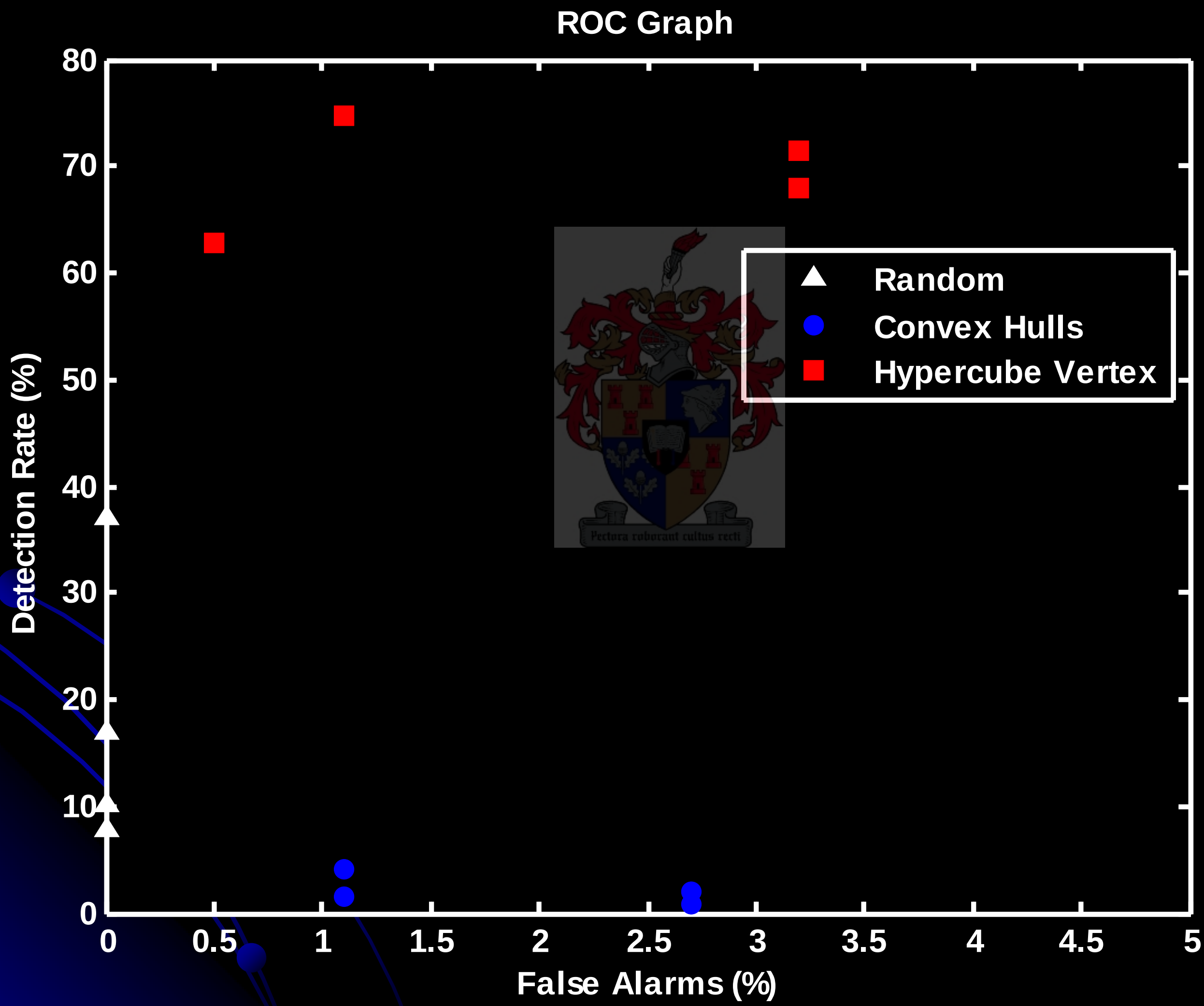


Self Space and Detection of Process Anomalies

- S → Attractor
- R → Detectors
- M → Data to be monitored
- Alarms marked with circles
- Anomalies Identified:
  - Drifting: 80% (76%)
  - Drifted: 63% (45%)
  - Error: 3.2% (0.5%)



Summary of different methods for generating detectors



Conclusions

- Immunocomputing is a new method for *adaptive* monitoring of *dynamic* processes.
- Large Dimensionality (100D)
  - Random
- Fewer dimensions (tested in 2D and 3D)
  - Hypercube Vertices
- Small as well as large changes detected in Steady State as well as Dynamic Systems

- Pro's:
  - Fast, effective
  - Online monitoring

- Con's:
  - Suitable Coding of Data an Issue
  - Generation of Detectors can also be an issue for large systems
  - Chaotic Systems

Recommendations

- For setting up AIS:
  - Carefully chose coding and embedding parameters
  - Define matching rule depending on requirements
  - Choose detectors according to their strengths
- Future study:
  - Hypermutation
  - Chaotic systems
  - Practical Implications

Acknowledgements

- Institutions:
  - University of Stellenbosch
  - NRF
- People:
  - Prof. Chris Aldrich
  - Gorden Jemwa
  - Friends, Family etc.

Questions?

Novel paradigms are proposed and accepted not necessarily for being faithful to their sources of inspiration, but for being useful and feasible.

DIAGNOSTIC MONITORING OF DYNAMIC PROCESS SYSTEMS USING ARTIFICIAL IMMUNE SYSTEMS

By: Charl Maree
Supervised by: Prof. Chris Aldrich

Department of Process Engineering,
University of Stellenbosch, South Africa

13 October 2006

PROCESS ENGINEERING
STELLENBOSCH-UNIVERSITY

Outline of Presentation

- Introduction
- The Natural Immune System
- The Artificial Immune System
  - Artificial Immune Algorithms
  - The Negative Selection Algorithm
- Methodology
  - Embedding the Self Space
  - Choice of Matching Rule
  - Generating Detectors
- Case Studies
  - Lokta-Volterra Process
  - Autocatalytic Reaction
  - Belousov-Zhabotinsky Reaction
- Conclusion
- Recommendations
- Acknowledgements
- Questions

Introduction

- Process anomaly detection is vital on modern process plants.
  - Approx. 70% of industrial accidents are caused by human errors.
  - Petrochemical industries in the USA alone lose an estimated 20 billion dollars annually.
  - Two of the most devastating chemical plant accidents, namely Union Carbide’s Bhopal (India) and Occidental Petroleum's Piper Alpha *might* have been prevented.
  - The Chernobyl disaster affected 3.7 million people across Europe.

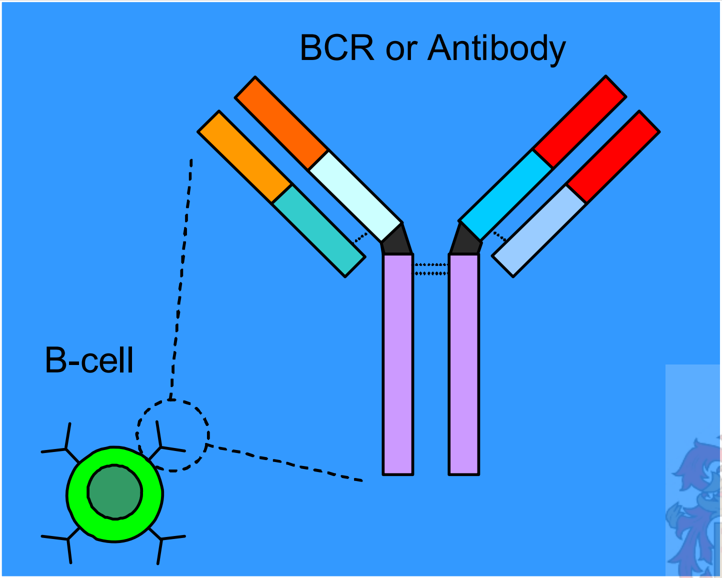
Introduction (cont.)

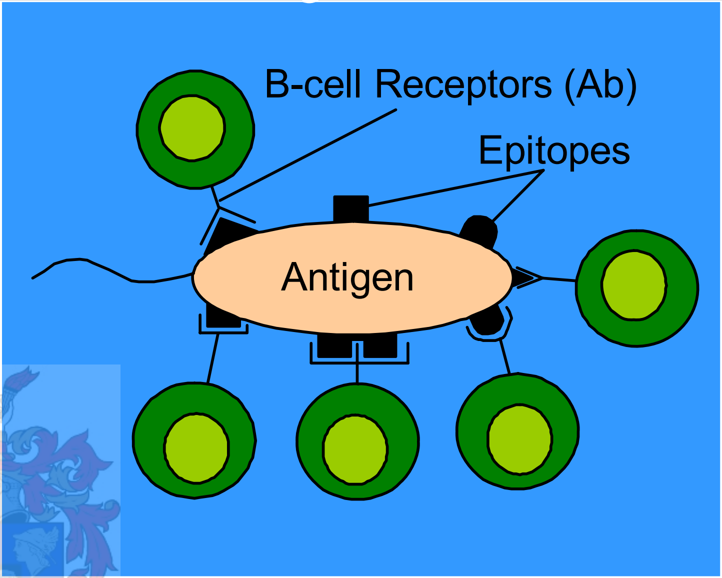
- Various established statistical methods for anomaly detection exist
- Unfortunately they are only defined when process operation meets certain assumptions, such as:
  - normal distributions of variables
  - linear correlations between variables
  - no auto- or cross-correlation in variables
- Immunocomputing is a promising analytical paradigm that has emerged very recently.

Introduction (cont.)

- Definition: A computational system based on the natural immune system, used for monitoring or diagnosis of process data.
- Why model the *immune system*?
  - Robustness
  - Adaptability
  - Multiple fault identifiability
  - Memory / Learning

Natural Immune Systems





- Recognition is based on the *complementarity* between *receptors* and *epitopes*.
- This analogy is applicable in a large variety of contexts:
    - Computer security systems
    - Pattern recognition
    - Optimisation
    - Detection of anomalies
    - **Process engineering!**

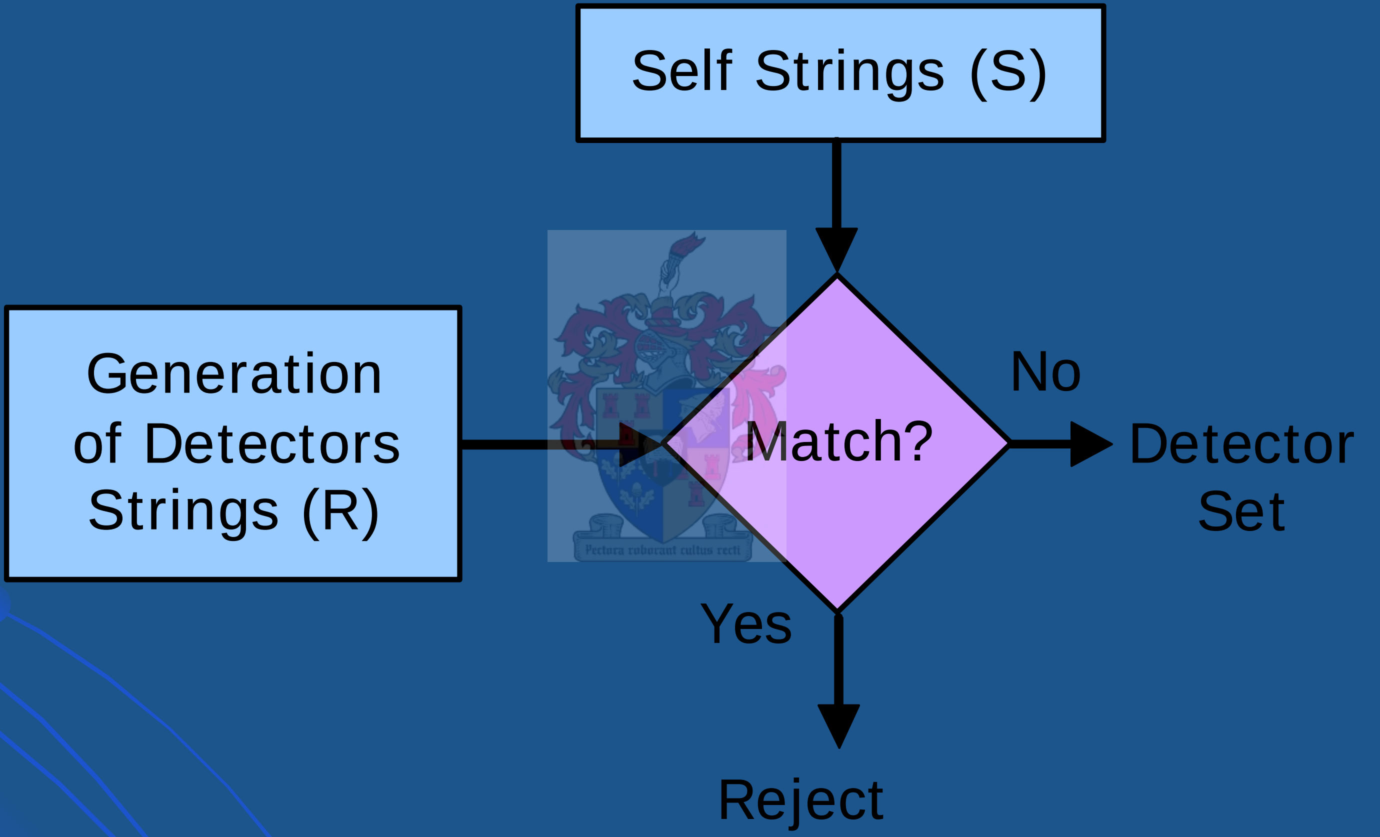
Artificial Immune Systems

- Do not attempt to *perfectly* match the natural immune system.
- Lymphocytes/Phagocytes → Detectors
- Antigens → Process anomalies
- Known Proteins → Historical process data (Self)
- Immune Response → Detector alarm

Artificial Immune Algorithms

- Sub-Classes in an AIS:
  - Positive and Negative Selection
  - Clonal Selection and Hypermutation
  - Artificial Immune Networks
  - Danger Theory

Negative Selection Algorithm: Censoring



Self Strings (S)

Generation of Detectors Strings (R)

Match?

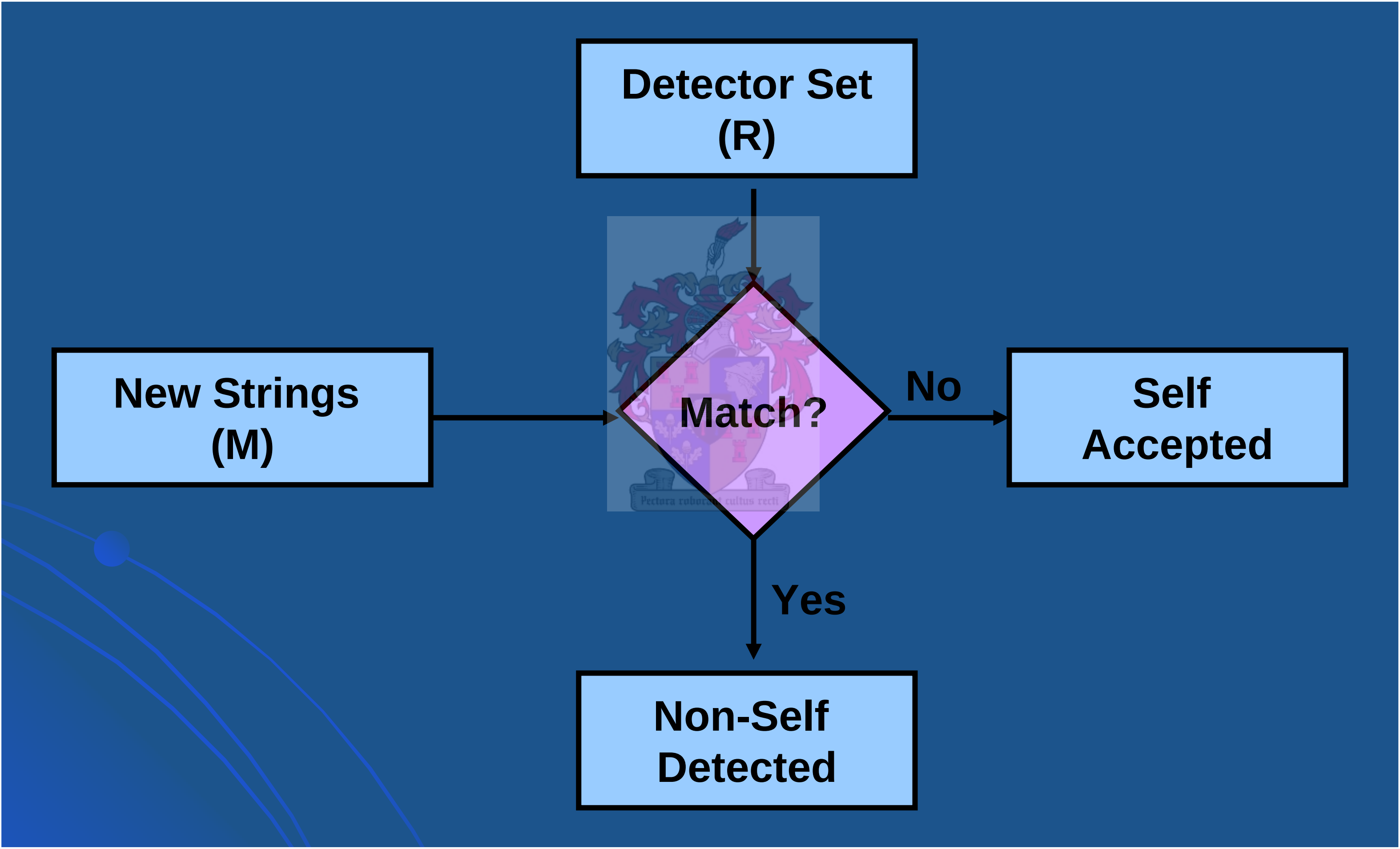
No

Detector Set

Yes

Reject

Negative Selection Algorithm: Monitoring



Embedding the Self Space

- Retrieve time series data
  - Real plant data
  - Simulated process
- Calculate the autocorrelation function R_L of the time series.
  - Choose the smallest L that satisfies $R_L = 0$

$$R_L(t) \equiv \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} f(L) f(t+L) dL$$

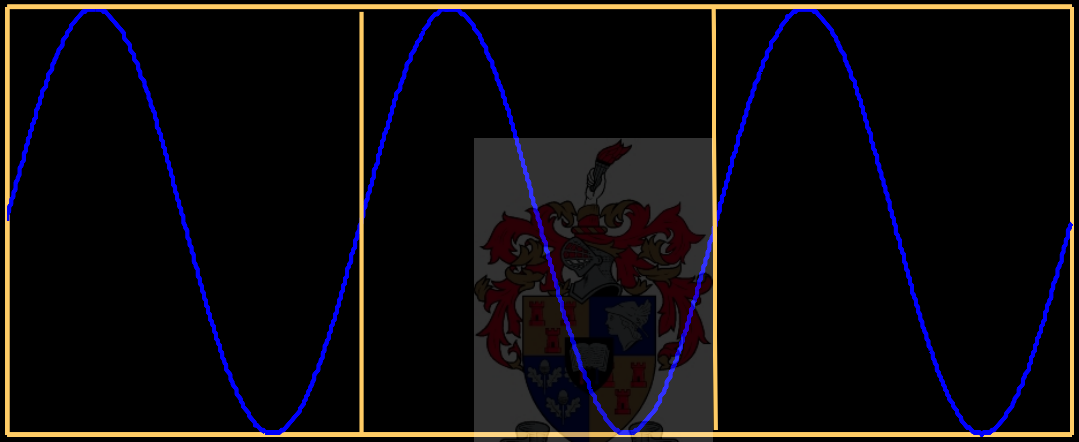
$$R_L \to -1 \qquad \textit{Alternating Series}$$

$$R_L \to \ \ 0 \qquad \textit{Random Series}$$

$$R_L \to +1 \qquad \textit{Smooth Series}$$

Embedding the Self Space (Cont.)

- "Split" time series into windows of length L.



- Stack windows to form matrix.
- For more than one time series, i.e. Temperature, Pressure and Concentration; multiple matrices are combined by joining rows

Embedding the Self Space (Cont.)

- Embedding is defined by the following mathematical equation

$$S_{n,ij} = T_{n,k}$$

$$where:$$

$$k = (i-1) \times L + j$$

$$n \quad is \quad the \quad number \quad of \quad time \quad series$$

$$S = \begin{bmatrix} S_{1,IJ} & S_{2,IJ} & \cdots & S_{N,IJ} \end{bmatrix}$$

$$where$$

$$I = 1,2,.., L$$

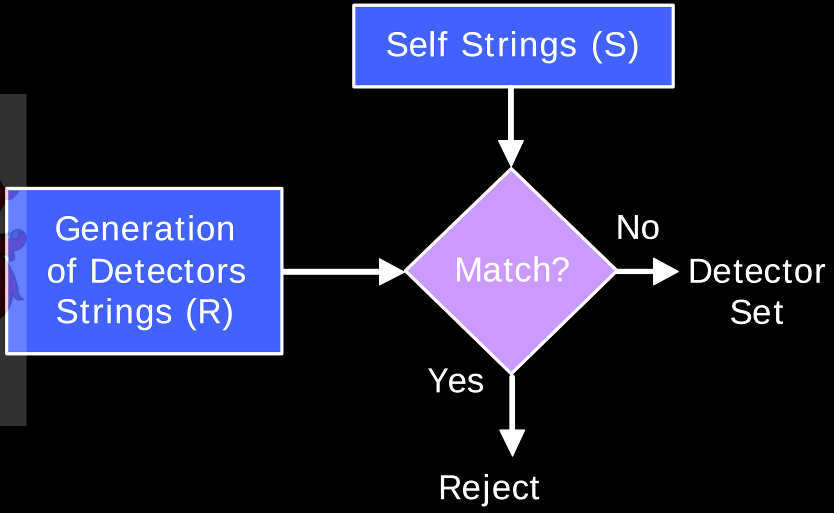
$$J = 1,2,.., \min(j_n)$$

Embedding the Self Space (Cont.)

- Apply PCA to determine the dimensionality of the self space.
- Approx. 90% of the original data variance should be captured.
- A smaller matrix (in dimensionality) is obtained which describes the self space.
- Embedding parameters:
  - Autocorrelation Lag (L)
  - Cumulative sum of PCA eigenvalues.

Choice of Matching Rule

- r-Contiguous bits (RCB)
- Euclidean Distance (ED)
- Other
  - Classic Hamming distance
  - Manhattan distance (aka Rectilinear)
  - Vector Cosine measure
  - L-norm distance
- Key Importance:
  - Ability to describe distance in the event space accurately



r-Contiguous Bits Matching Rule

- Best suited for a binary alphabet
- Example:
  - Let Ab = [1 1 0 1 1 0 0]
  - Let Ag = [1 0 0 1 1 0 1]
  - match(Ab,Ag) = true for r <= 4

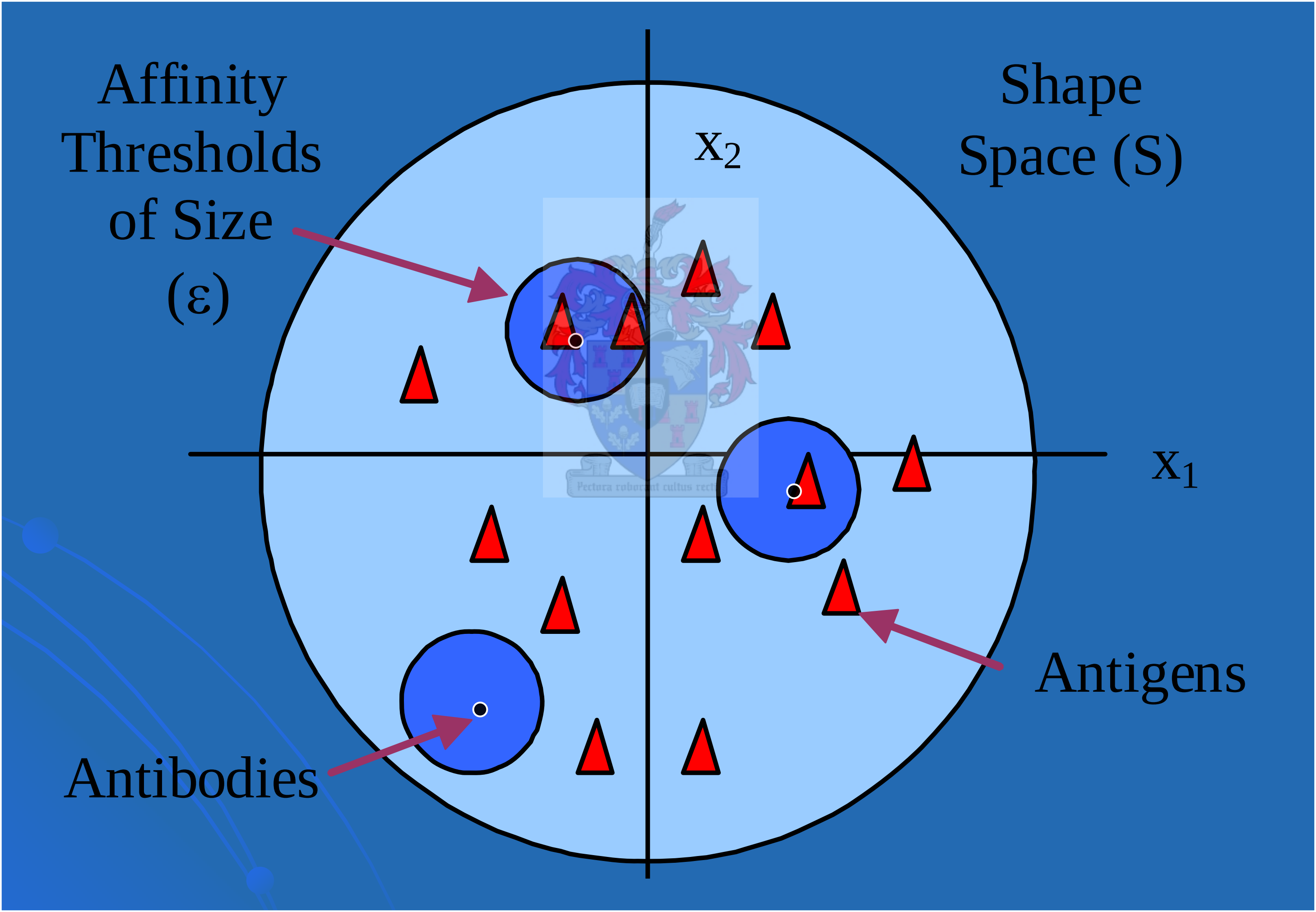
Euclidean Distance Matching Rule

- Suited for a larger alphabets, i.e. real data
- Affinity is related to distance:

$$D = \sqrt{\sum_{i=1}^{L} (Ab_i - Ag_i)^2}$$

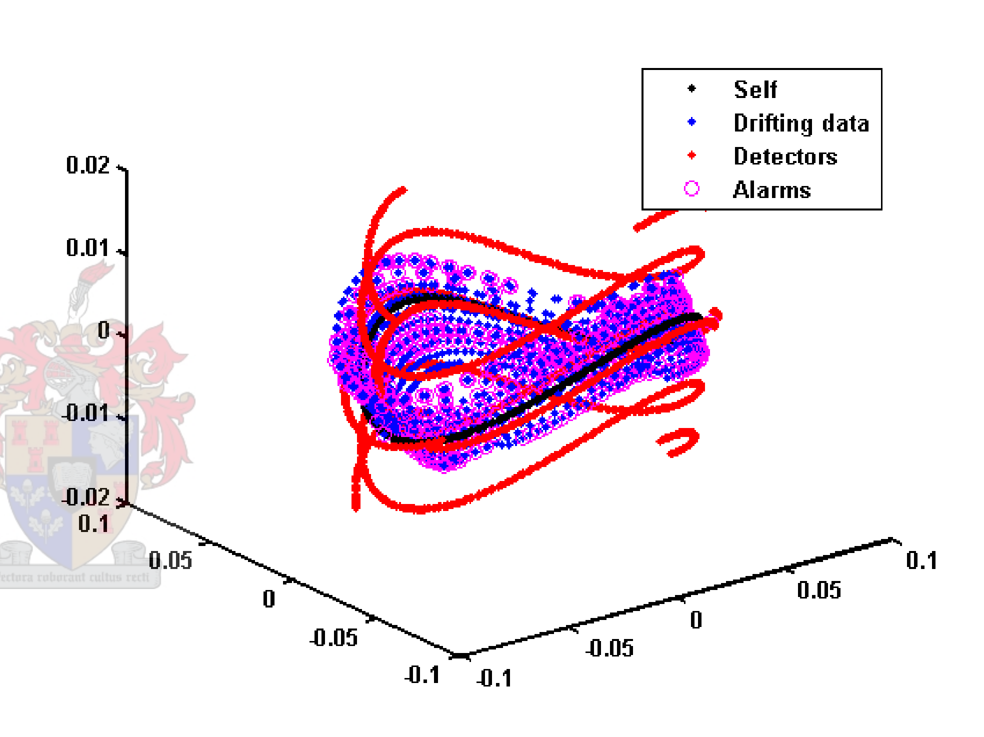
- match(Ab,Ag) = true if D < threshold

Affinity Threshold in the Ab-Ag Shape Space


Affinity Thresholds of Size (ε)
x_2
Shape Space (S)
x_1
Antigens
Antibodies
Pectora roborant cultus recti

Generating Detectors

- Random
- Convex Hulls
- Hypercube Vertices
- Important factors:
  - Time to generate *x* detectors
  - Ability of detectors to identify process anomalies
  - False alarms generated



Random Detectors

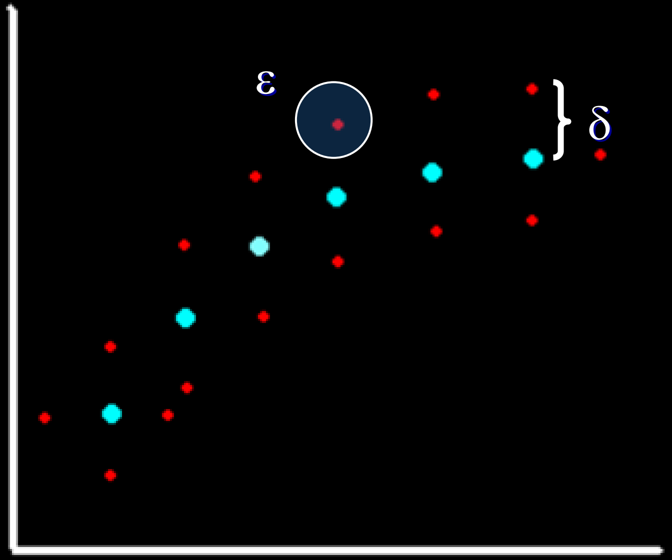
- Generate a random string containing coordinates in the n-Dimensional shape space.
- Test for a match with any of the points in the self space.
- If match = false, add the string to the detector pool.

Convex Hulls

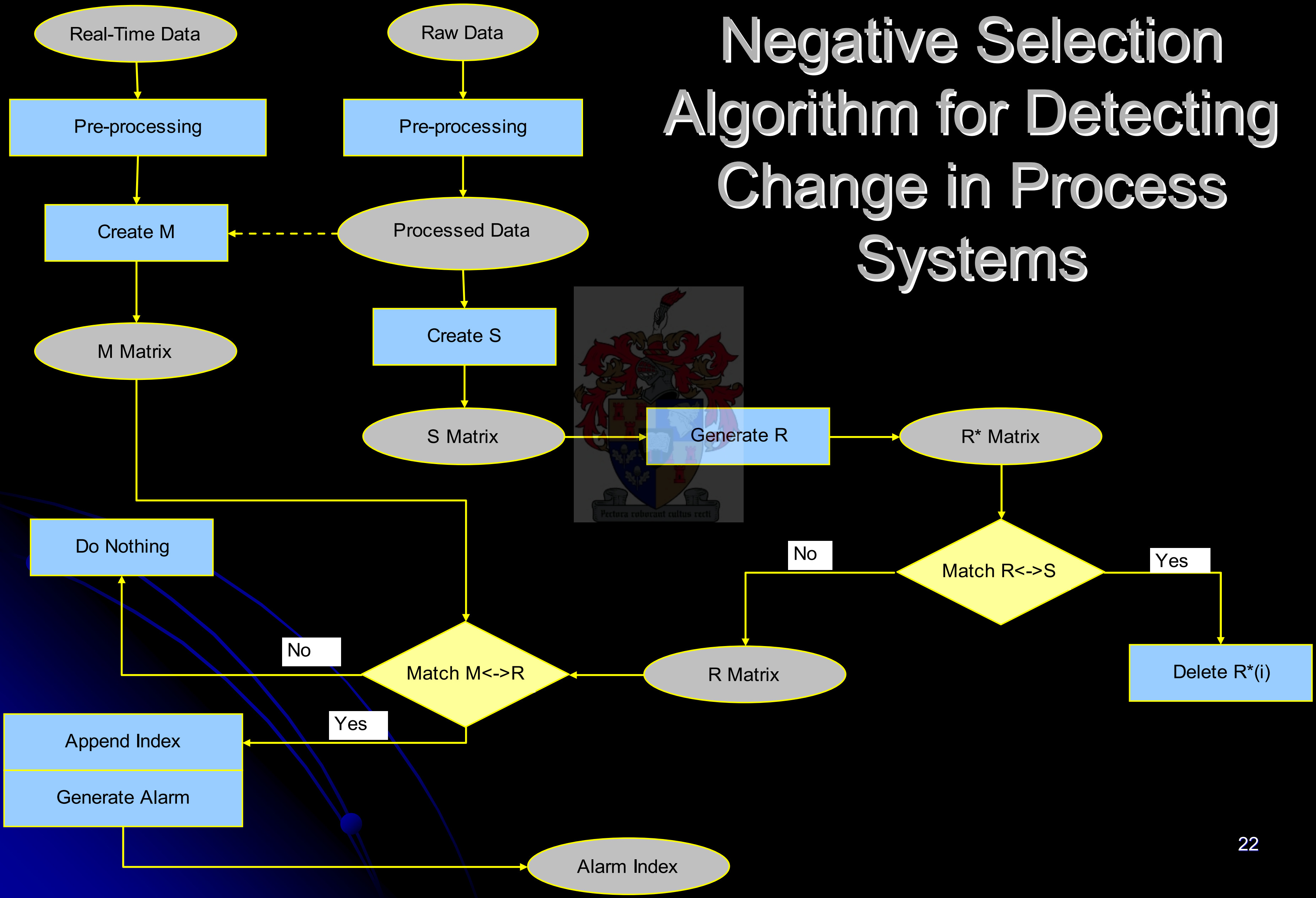
- Identify the boundary of the self space
  - Add Gaussian noise of size epsilon (ε)
  - Fit the convex hulls around the self space
  - Remove a layers one at a time until inner boundaries are defined
- Position detectors on both outer and inner boundaries

Hypercube Vertices Method

- Position detectors on vertices of a hypercube around each self event
- Remove unwanted detectors
- Two Parameters:
  - Epsilon (ε)
    - Specifies sensitivity
    - Sensor radius around detector
  - Delta (δ)
    - Specifies flexibility
    - Minimum distance to nearest self string



Negative Selection Algorithm for Detecting Change in Process Systems



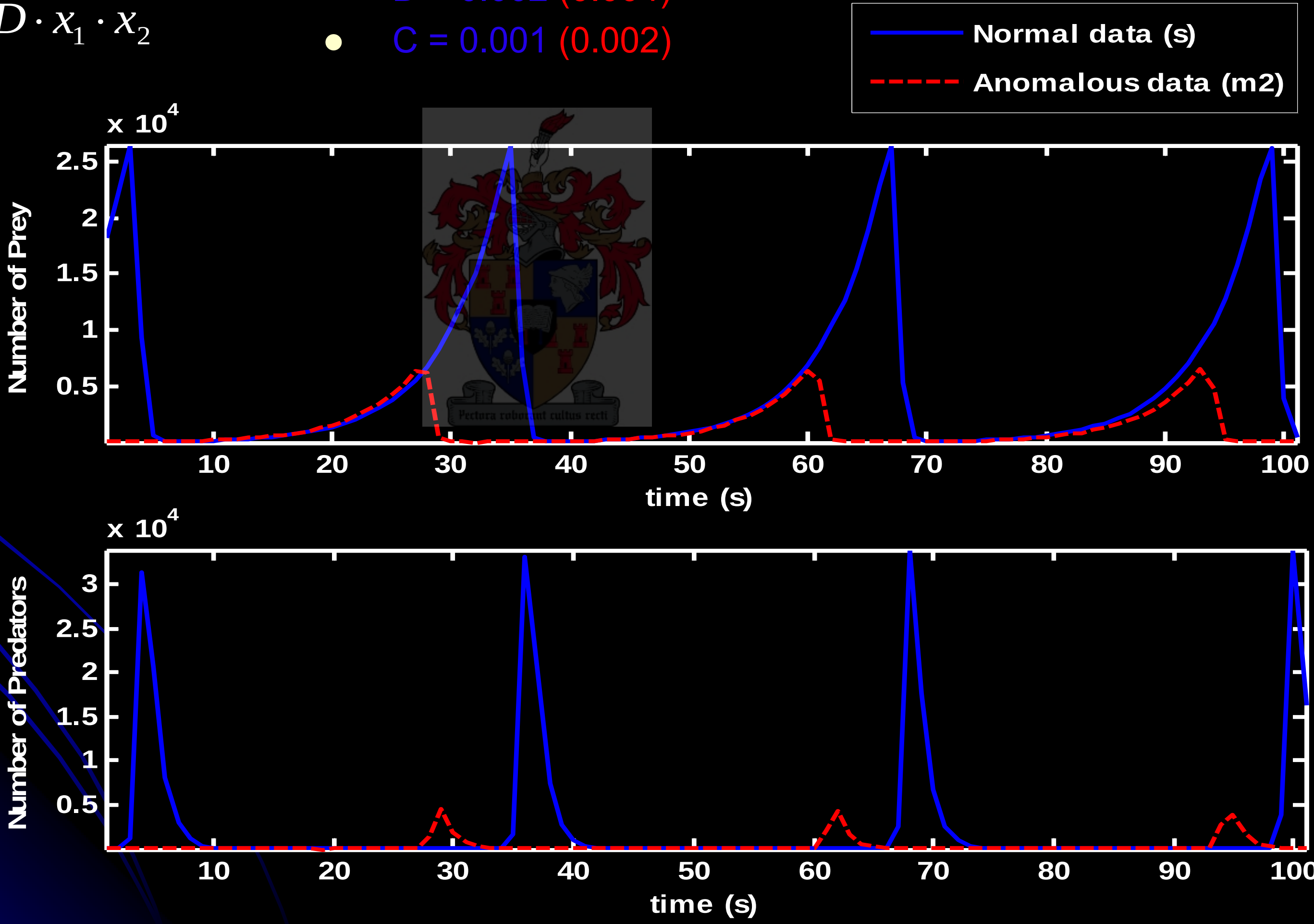
Simulation 1: Lokta-Volterra Process

$$\frac{dx_1}{dt} = k_1 \cdot x_1 - C \cdot x_1 \cdot x_2$$

$$\frac{dx_2}{dt} = -k_2 \cdot x_2 + D \cdot x_1 \cdot x_2$$

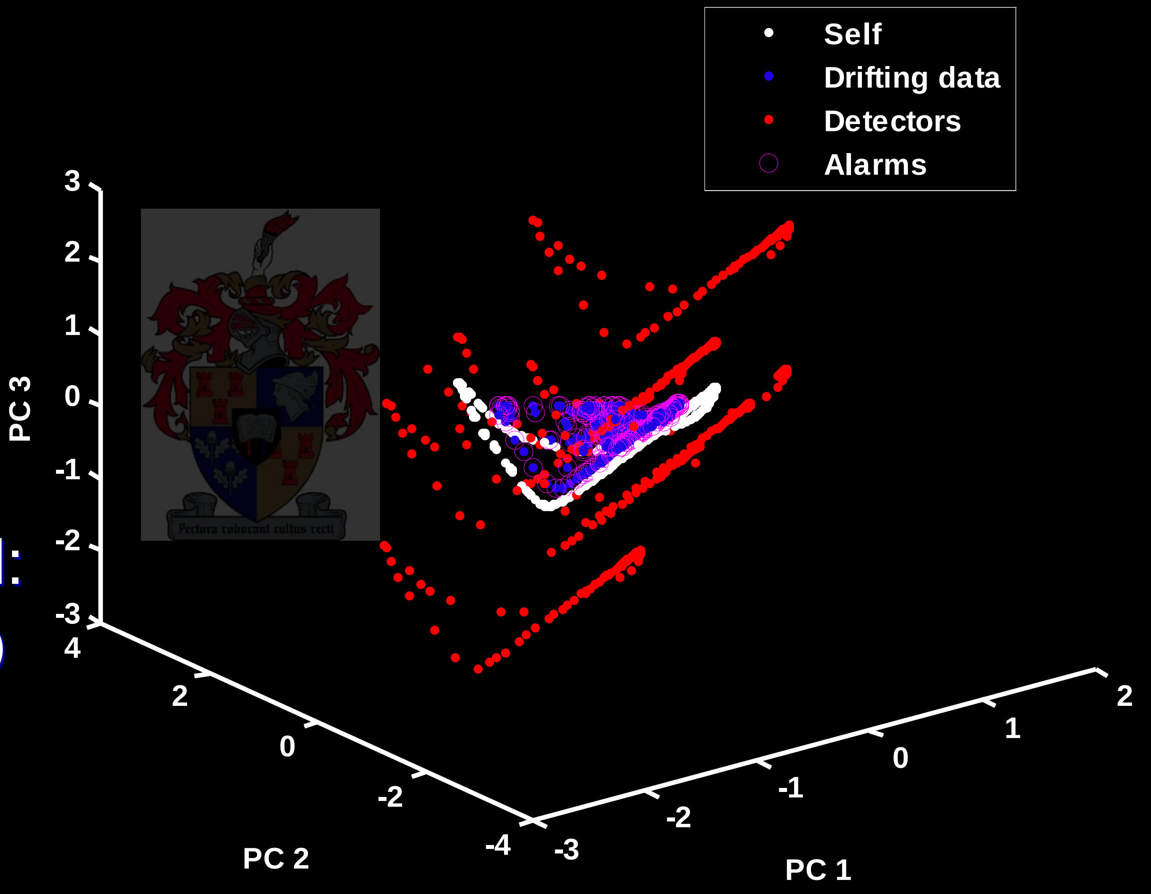
where:

- $k_1 = 2$
- $k_2 = 10$
- D = 0.002 (0.004)
- C = 0.001 (0.002)

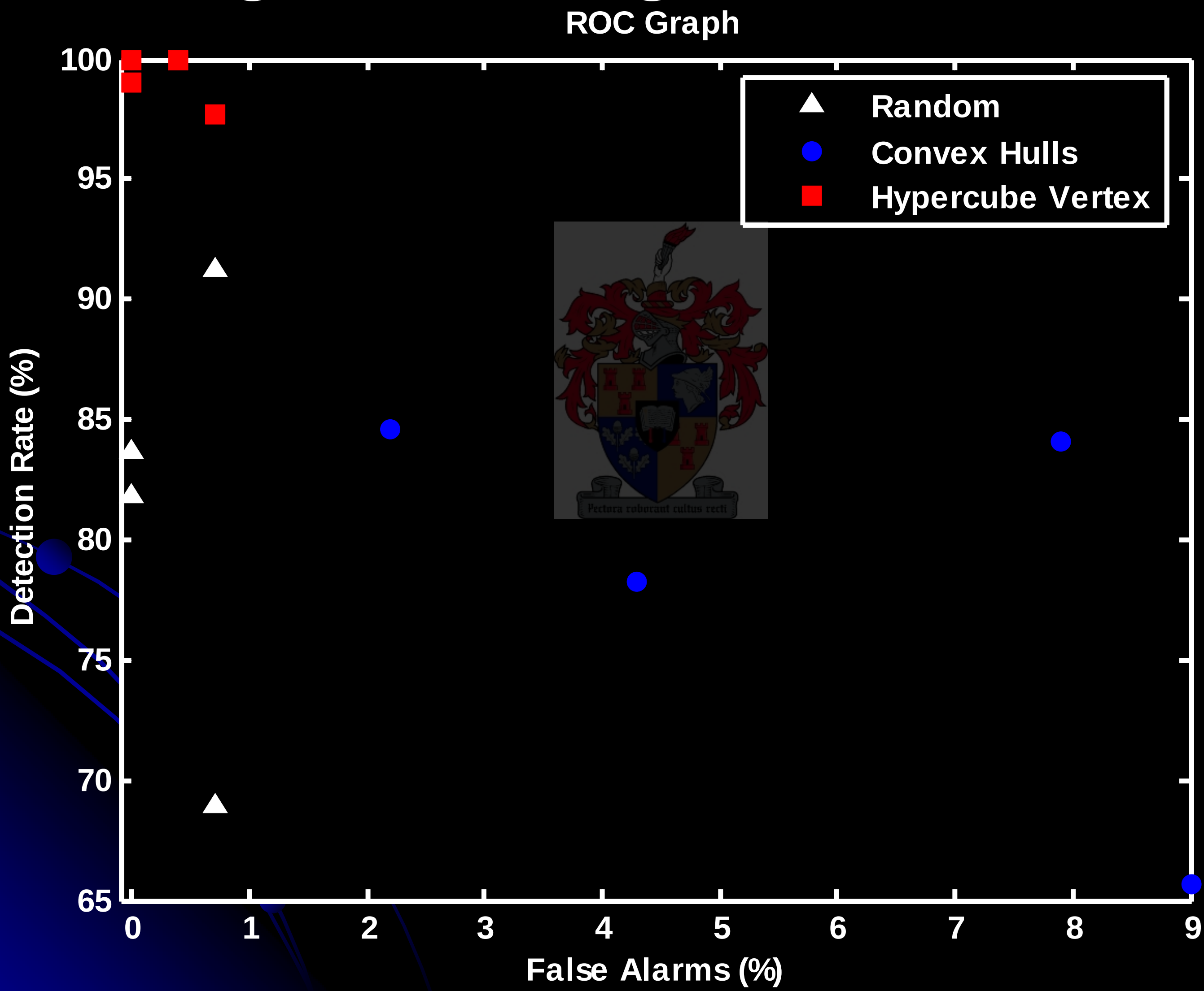


Self Space and Detection of Process Anomalies

- S → Attractor
- R → Detectors
- M → Data to be monitored
- Alarms marked with circles
- Anomalies Identified:
  - Drifting: 100% (98%)
  - Drifted: 100% (96%)
  - Error: 0% (0.7%)



Summary of different methods for generating detectors



Simulation 2: Autocatalytic Process

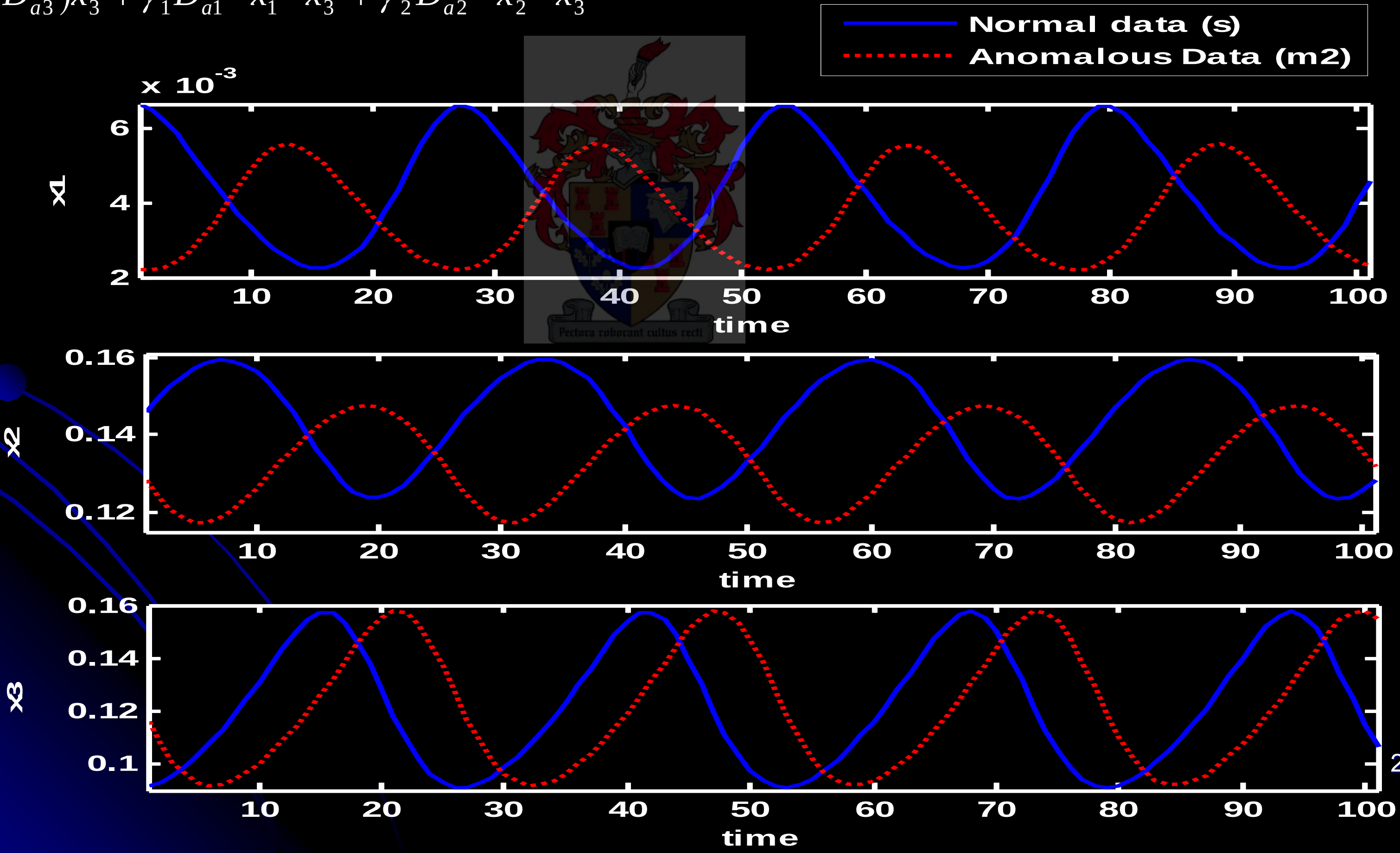
$$\frac{dx_1}{d\tau} = 1 - x_1 - D_{a1} \cdot x_1 \cdot x_3^2$$

$$\frac{dx_2}{d\tau} = 1 - x_2 - D_{a2} \cdot x_2 \cdot x_3^2$$

$$\frac{dx_3}{d\tau} = 1 - (1 + D_{a3})x_3 + \gamma_1 D_{a1} \cdot x_1 \cdot x_3^2 + \gamma_2 D_{a2} \cdot x_2 \cdot x_3$$

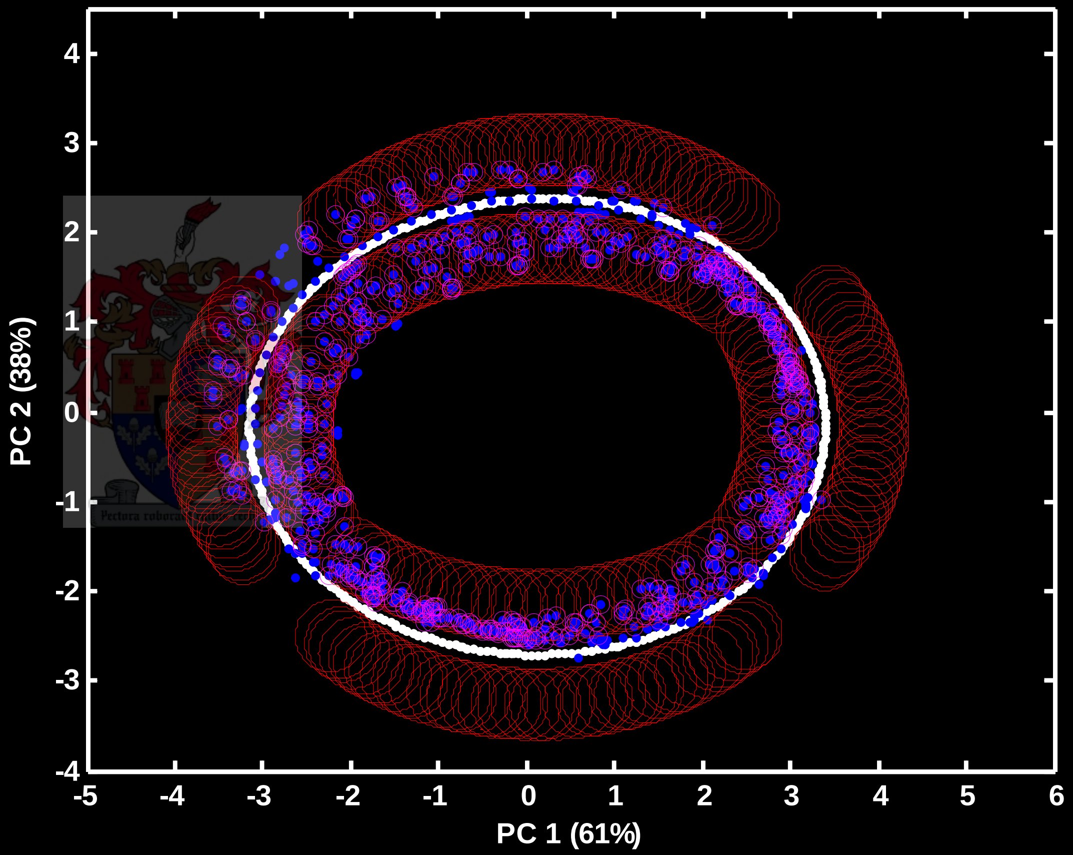
$D_{a1} = 18000$ $D_{a2} = 400$ $D_{a3} = 80$

$\gamma_1 = 3.5$ (3.7) $\gamma_2 = 6.2$ (6.4)

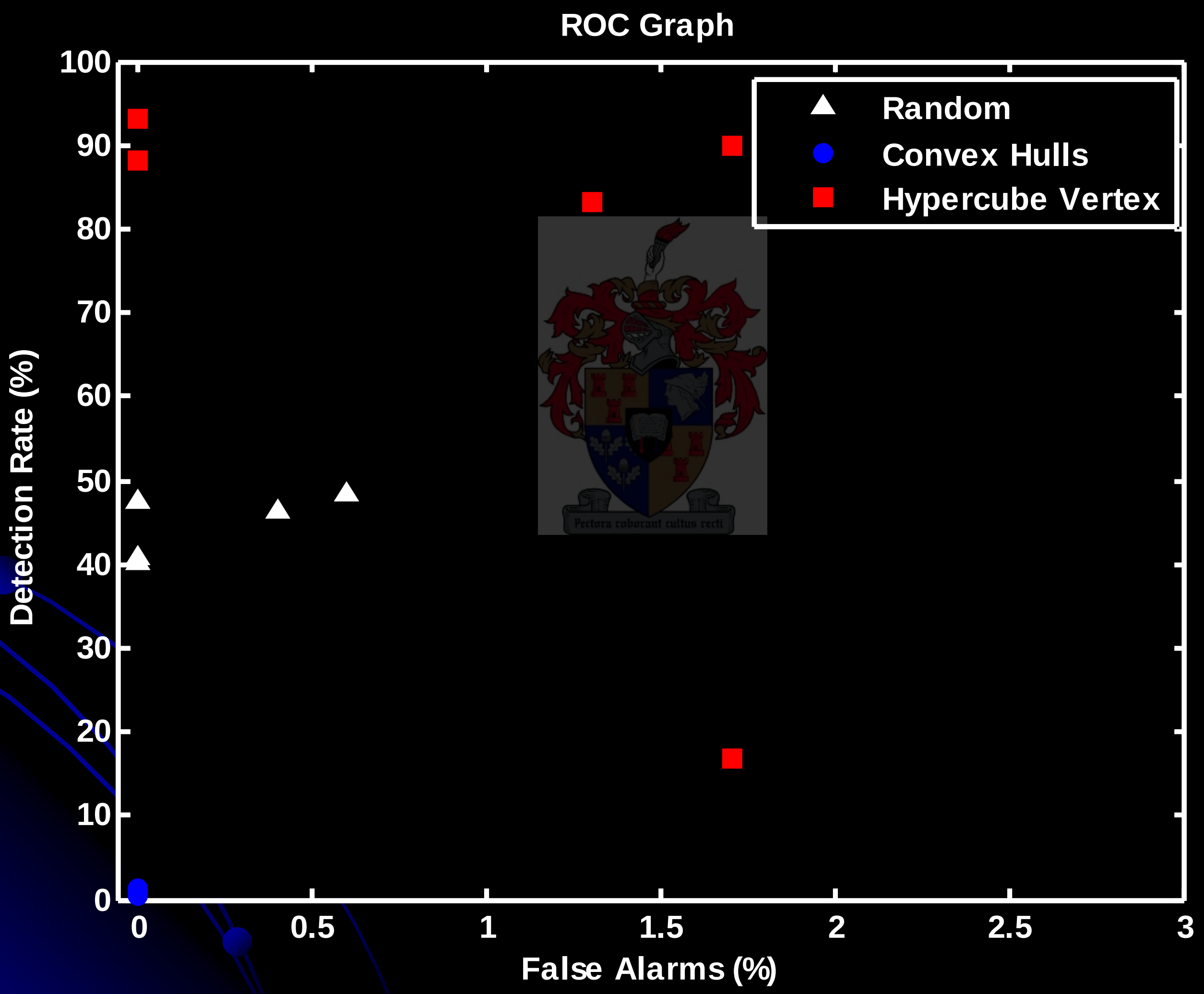


Self Space and Detection of Process Anomalies

- S → Attractor
- R → Detectors
- M → Data to be monitored
- Alarms marked with circles
- Anomalies Identified:
  - Drifting: 100% (98%)
  - Drifted: 80% (72%)
  - Error: 1.7% (0.4%)



Summary of different methods for generating detectors



Simulation 3: Belousov-Zhabotinsky Reaction

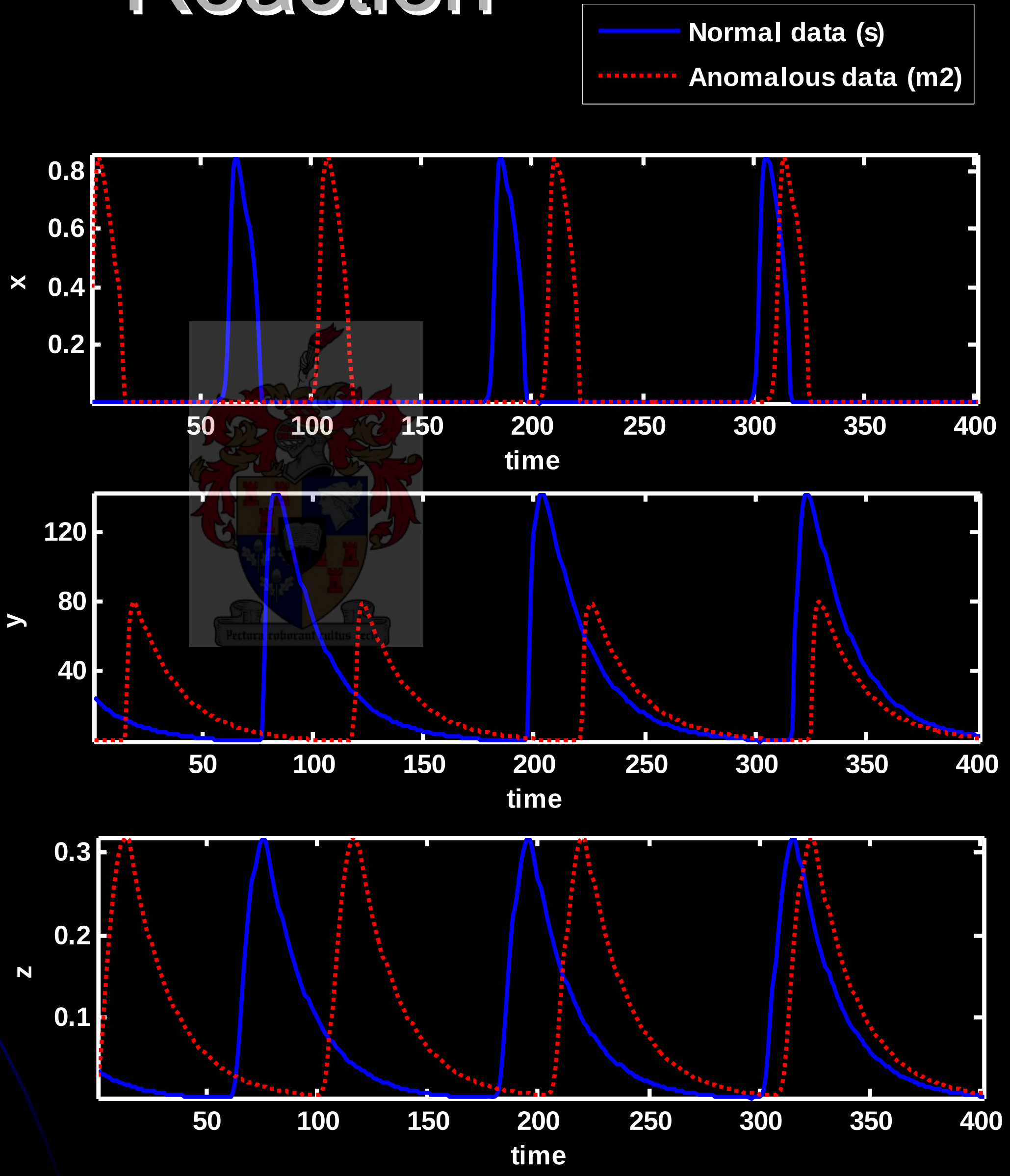
$$\frac{dx}{d\tau} = \frac{1}{c_1}\left[qx - xy + x(1-x)\right]$$

$$\frac{dy}{d\tau} = \frac{1}{c_2}\left[-qy - xy + fz\right]$$

$$\frac{dz}{d\tau} = x - z$$

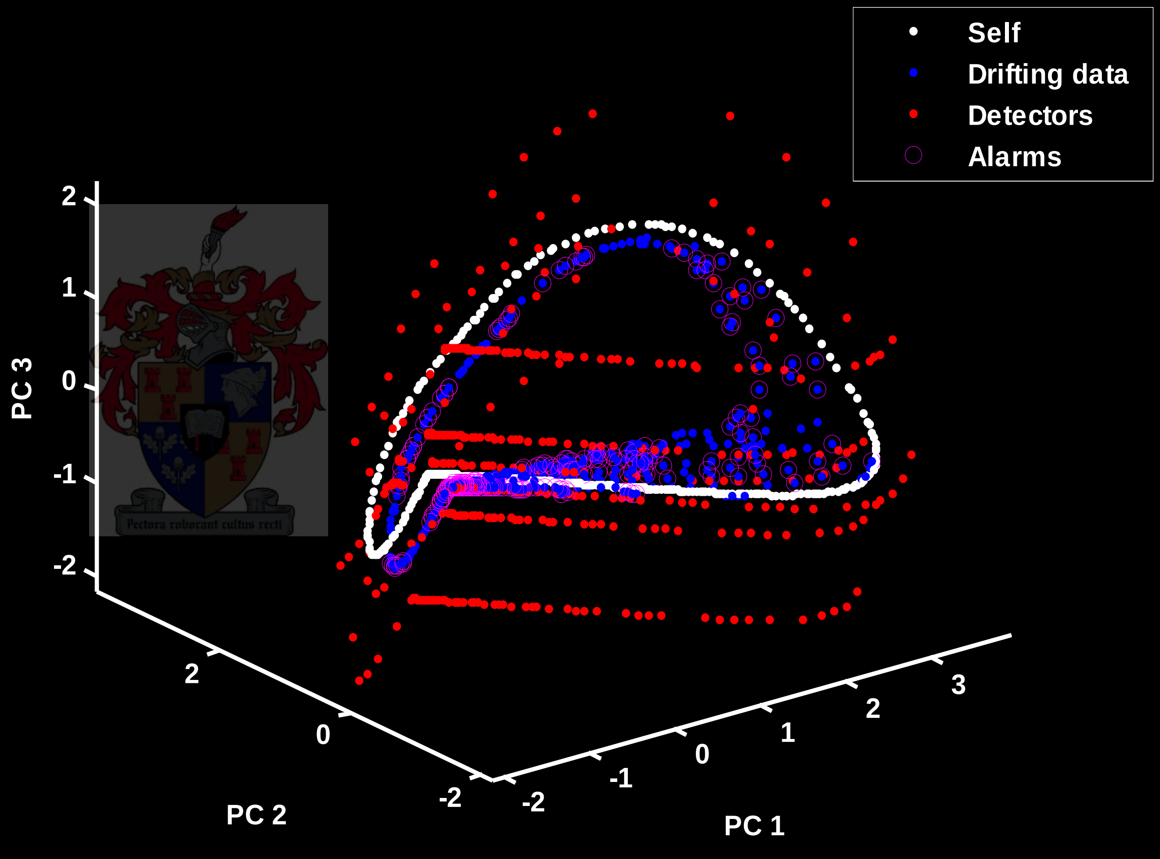
Where:

- $c_1 = 0.04$
- $c_2 = 0.0002$
- $q = 0.0008$ (0.0016)

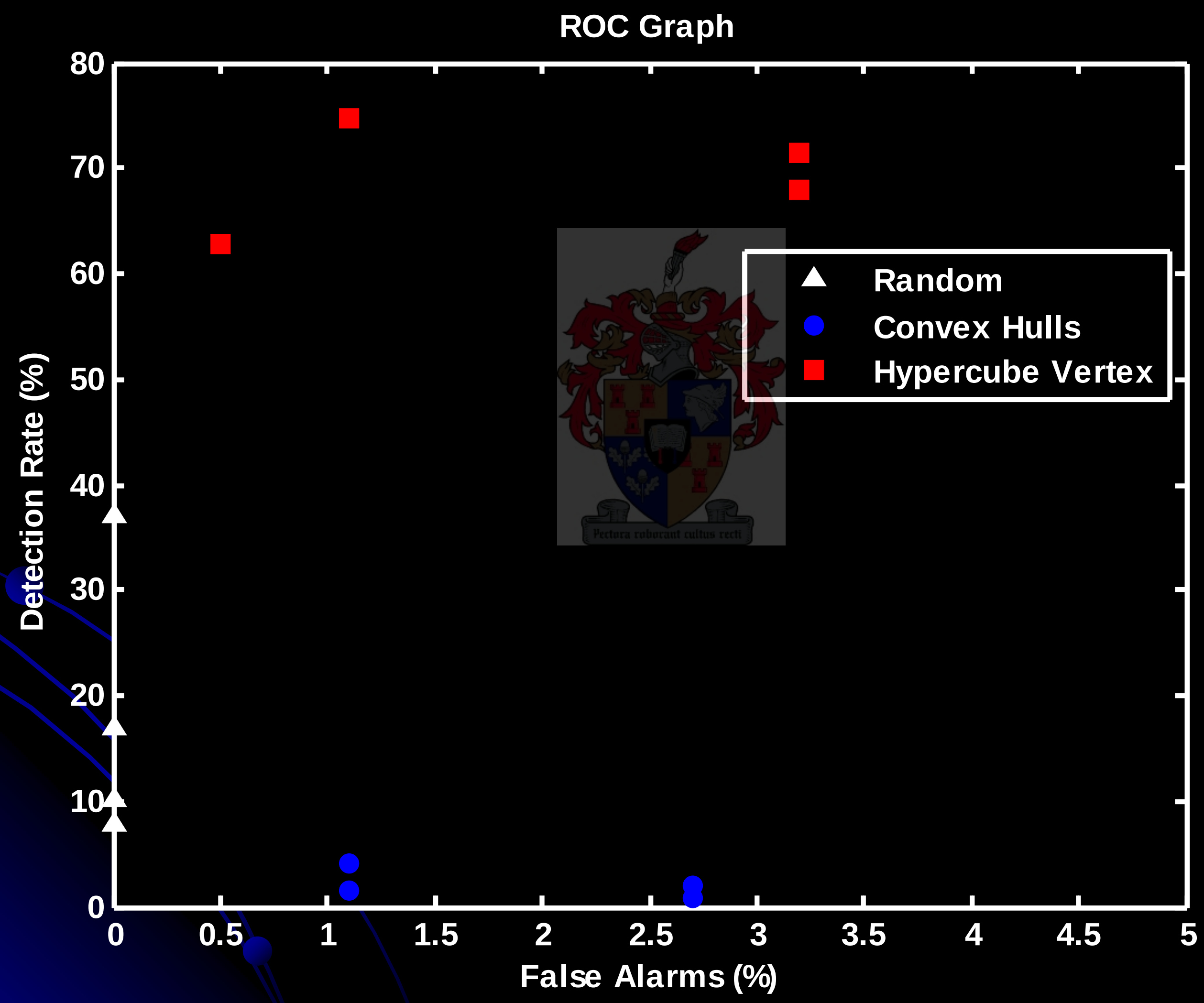


Self Space and Detection of Process Anomalies

- S → Attractor
- R → Detectors
- M → Data to be monitored
- Alarms marked with circles
- Anomalies Identified:
  - Drifting: 80% (76%)
  - Drifted: 63% (45%)
  - Error: 3.2% (0.5%)



Summary of different methods for generating detectors



Conclusions

- Immunocomputing is a new method for *adaptive* monitoring of *dynamic* processes.
- Large Dimensionality (100D)
  - Random
- Fewer dimensions (tested in 2D and 3D)
  - Hypercube Vertices
- Small as well as large changes detected in Steady State as well as Dynamic Systems

- Pro's:
  - Fast, effective
  - Online monitoring

- Con's:
  - Suitable Coding of Data an Issue
  - Generation of Detectors can also be an issue for large systems
  - Chaotic Systems

Recommendations

- For setting up AIS:
  - Carefully chose coding and embedding parameters
  - Define matching rule depending on requirements
  - Choose detectors according to their strengths
- Future study:
  - Hypermutation
  - Chaotic systems
  - Practical Implications

Acknowledgements

- Institutions:
  - University of Stellenbosch
  - NRF
- People:
  - Prof. Chris Aldrich
  - Gorden Jemwa
  - Friends, Family etc.

Questions?

Novel paradigms are proposed and accepted not necessarily for being faithful to their sources of inspiration, but for being useful and feasible.