Fault Detection and Isolation of Wind Turbines using Immune System Inspired Algorithms

Esmaeil AliZadeh

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montréal, Québec, Canada

November 2016

CONCORDIA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Esmaeil AliZadeh

Entitled: Fault Detection and Isolation of Wind Turbines using Immune System Inspired Algorithms

and submitted in partial fulfilment of the requirements for the degree of

Master of Applied Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Dr. Skonieczny, Chair

_____ Dr. M. Paraschivoiu, External Examiner

_____ Dr. S. Shih, Examiner

_____ Dr. K. Khorasani, Co-Supervisor

_____ Dr. N. Meskin, Co-Supervisor

Approved by: _____

Dr. W. E. Lynch, Chair

Department of Electrical and Computer Engineering

_____ 20 _____

Dr. A. Asif

Dean, Faculty of Engineering and Computer Science

ABSTRACT

Fault Detection and Isolation of Wind Turbines using Immune System Inspired Algorithms

Esmaeil AliZadeh

Recently, the research focus on renewable sources of energy has been growing intensively. This is mainly due to potential depletion of fossil fuels and its associated environmental concerns, such as pollution and greenhouse gas emissions. Wind energy is one of the fastest growing sources of renewable energy, and policy makers in both developing and developed countries have built their vision on future energy supply based on and by emphasizing the wind power. The increase in the number of wind turbines, as well as their size, have led to undeniable care and attention to health and condition monitoring as well as fault diagnosis of wind turbine systems and their components.

In this thesis, two main immune inspired algorithms are used to perform Fault Detection and Isolation (FDI) of a Wind Turbine (WT), namely the *Negative Selection Algorithm (NSA)* as well as the *Dendritic Cell Algorithm (DCA)*.

First, an NSA-based fault diagnosis methodology is proposed in which a hierarchical bank of NSAs is used to detect and isolate both individual as well as simultaneously occurring faults common to the wind turbines. A smoothing moving window filter is then utilized to further improve the reliability and performance of the proposed FDI scheme. Moreover, the performance of the proposed scheme is compared with the state-of-the-art

data-driven technique, namely Support Vector Machine (SVM) to demonstrate and illustrate the superiority and advantages of the proposed NSA-based FDI scheme. Finally, a nonparametric statistical comparison test is implemented to evaluate the proposed methodology with that of the SVM under various fault severities.

In the second part, another immune inspired methodology, namely the Dendritic Cell Algorithm (DCA) is used to perform online sensor fault FDI. A noise filter is also designed to attenuate the measurement noise, resulting in better FDI results. The proposed DCA-based FDI scheme is then compared with the previously developed NSA-based FDI scheme, and a nonparametric statistical comparison test is also performed.

Both of the proposed immune inspired frameworks are applied to a well-known wind turbine benchmark model in order to validate the effectiveness of the proposed methodologies.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisors Dr. Khorasani and Dr. Meskin for their support of my master thesis and research, and also for their immense knowledge and remarks throughout the learning process of this master thesis. This thesis could not be accomplished without their guidance and insightful comments. I am grateful to Dr. Meskin for introducing me to artificial immune systems all those years ago.

I would also like to thank my colleagues and fellow labmates at Concordia University: Dr. Amir Baniamerian, Mahsa Khoshab, Maria Enayat, Niusha Modabbernia, Niloofar Ashtiani, Yanyan Shen, Maryam Abdollahi, Arefeh Amrollahi, Amin Salar, and Bahar Pourbabaee for all the stimulating discussions and exchanges, as well as for all the fun and nice time we have had throughout these years. Moreover, I would also like to thank my friends Ebrahim Rashidi and Halimeh Madani.

Last but not the least, I would like to thank my beloved ones in particular my parents, my sister, and my brothers for their unconditional love and inspiration throughout my life. None of this would have been possible without their encouragement. My special thanks go to my brother, Dr. Hussain Alizadeh, who has supported me wholeheartedly throughout my studies, and also for his guidance in the immunological perspective of my thesis.

List of Publications

[1] E. Alizadeh, N. Meskin, and K. Khorasani, "A Dendritic Cell Immune System Inspired Scheme for Sensor Fault Detection and Isolation of Wind Turbines," *IEEE Transactions on Industrial Informatics*, 2016, *(under review)*.

[2] E. Alizadeh, N. Meskin, and K. Khorasani, "A Negative Selection Immune System Inspired Methodology for Fault Diagnosis of Wind Turbines," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1 – 15, 2016.

[3] E. Alizadeh, N. Meskin, and K. Khorasani, "A Sensor Fault Detection and Isolation Strategy by using a Dendritic Cell Algorithm," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, Hungary, Oct. 9 – 12, 2016.

Table of Contents

List of Figures

List of Tables

Nomenclature

Wind Turbine

$\beta_{i,\,\mathrm{ref}}$	Pitch reference angle for blade i	[°]
β_{i,m_j}	Pitch angle value of the blade i sensor j	[°]
B_g	Viscous friction of the generator shaft	[rad/s]
B_r	Viscous friction of the rotor shaft	[rad/s]
B_{DT}	Torsional damping coefficient of the drive-train	[Nm · s/rad]
C_p	Power coefficient of the wind turbine	[–]
η_g	Generator efficiency	[–]
η_{DT}	Efficiency of the drive-train	[–]
J_g	Moment of inertia of the generator shaft	[Kg · m^2]
J_r	Moment of inertia of the rotor shaft	[Kg · m^2]

K_{DT}	Torsional stiffness coefficient of the drive-train [Nm/rad]
$\lambda(t)$	Tip-Speed Ratio [–]
$v_{\beta}(t)$	External measurement noise added to $\beta(t)$ [°]
$v_{\omega_g}(t)$	External measurement noise added to $\omega_g(t)$ [rad/s]
$v_{\omega_r}(t)$	External measurement noise added to $\omega_r(t)$ [rad/s]
$v_{P_g}(t)$	External measurement noise added to $P_g(t)$ [W]
N_g	Gear ratio of the drive-train gearbox [–]
ω_n	Natural frequency of the second-order pitch actuator model [rad/s]
$\omega_{g,m}$	Generator speed measurement [rad/s]
$\omega_{r,m}$	Rotor speed measurement [rad/s]
R	Rotor radius (length of blades) [m]
τ_g	Generator torque [N · m]
τ_r	Aerodynamic torque acting on rotor blades [N · m]
θ_{Δ}	Torsional angle of the drive-train [°]
T_{gc}	Time constant of the first-order approximation of the generator and converter dynamics [s]

$v_{\mathrm{w}}(t)$	Overall wind speed	[m/s]
ζ	Damping ratio of the second-order pitch actuator model	[–]

Other Symbols

D	Detector set (NSA)
S	Self set (NSA)
Γ	Euclidean distance
$\hat{S}_{m_j}(t)$	Filtered measurement of the sensor $S_{m_j}(t)$
$\mathcal{F}$	F-score (also known as F-measure or F_1-score)
$r(t)$	Residual signal generated from an NSA filter at time t
$r^{S}_{\mathrm{DCA},m_j}(t)$	DCA residual signal corresponding to the sensor S_{m_j}
$r^{S}_{\mathrm{NSA},m_j}(t)$	NSA residual signal corresponding to the sensor S_{m_j}
R_{self}	Self-radius of hyperspheres (NSA)
DCA^{S}	DCA filter designed for the sensor S
$\mathrm{DS}^{S}_{m_j}(t)$	DS signal corresponding to the sensor S_{m_j} at time t
NSA^{S}	NSA filter designed for the sensor S

$\mathrm{PS}^{S}_{m_j}(t)$	PS signal corresponding to the sensor S_{m_j} at time t	
$\mathrm{SS}^{S}_{m_j}(t)$	SS signal corresponding to the sensor S_{m_j} at time t	
T	Time constant of the linear first-order dynamics	[s]

List of Acronyms

AIS Artificial Immune System

BPS Blade and Pitch System

CI Computational Intelligence

CSM Co-Stimulatory Molecule

DAT DAnger Theory

DC Dendritic Cell

DCA Dendritic Cell Algorithm

dDCA deterministic DCA

DR Detection Rate

DS Danger Signal

DT Drive-Train

FA False Alarm rate

FDI Fault Detection and Isolation

FN False Negative

FP False Positive

FTC Fault Tolerant Control

HIS Human Immune System

HSS high-speed shaft

IC Inflammatory Cytokine (Inflammatory Signal)

iDC Immature DC

IL-10 interleukin-10

IL-12 interleukin-12

IS Immune System

LSS low-speed shaft

mDC Mature DC

NS Negative Selection

NSA Negative Selection Algorithm

PAMP Pathogen-Associated Molecular Pattern

PS PAMP Signal

smDC Semi-mature DC

SNS Self-Non-Self

SS Safe Signal

SVM Support Vector Machine

TN True Negative

TP True Positive

WT Wind Turbine

Chapter 1

Introduction

1.1 Motivation

The need for renewable energy sources has been growing significantly, especially due to the energy crisis and its associated environmental concerns. Wind energy is one of the fastest growing source of renewable energy for power generation in past years. Consequently, the Wind Turbine (WT) industry is getting bigger as more large-scale WTs are being used. On the other hand, high operational and maintenance costs are the major economic constraints in the WT industry. These concerns have made the investigation into fault diagnosis of WT systems an imperative and active area of research. Therefore, this thesis explores the Fault Detection and Isolation (FDI) of various fault scenarios in WT systems.

To address the task of FDI, Computational Intelligence (CI) methods are considered. The use of CI to address complex real-world problems has been increasing mainly due to the limited use of model-based methods, particularly when a process is too complicated for mathematical reasoning, or the process has a stochastic nature. Among various CI techniques, Artificial Immune System (AIS) is rapidly emerging in many fields such as classification, pattern recognition, computer security problems, anomaly detection, and *etc.* The main reason for the growing research interest in AIS is due to its significant information processing capabilities such as classification, pattern recognition, optimization, learning, memory, and distributed parallel processing. In this thesis, two AIS algorithms, namely the Negative Selection Algorithm (NSA) and Dendritic Cell Algorithm (DCA)) are employed to perform the FDI task within the WT system.

1.2 Fault Detection and Isolation (FDI)

Fault Detection and Isolation (FDI) problem has been the subject of many research activities due to its importance in industry. According to Isermann [1], a *fault* is defined as follows:

"A fault is an unpermitted deviation of at least one characteristics property (feature) of the system from the acceptable, usual, standard condition."

It should be noted that there is a difference between a *fault* and a *failure*. Unlike a *fault* that causes a deviation in a system characteristic from its nominal situation, a *failure* is

"a *permanent* interruptions of a system's ability to perform a required function under specified operating condition [2]."

In general, three fault types can occur in a plant (shown in Figure 1.1) as follows [3]:

1. *Sensor faults*,
2. *Actuator faults*,
3. *Component (also known as system, plant, or process) faults.*

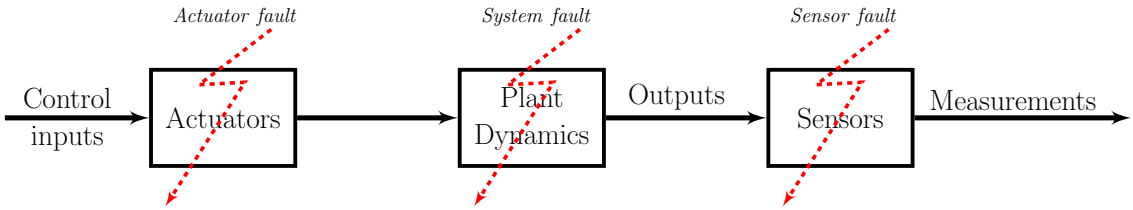


Figure 1.1: Different types of fault.

The task of fault detection as well as fault isolation can be defined as below:

- **Fault detection:** indicates if there is a fault in the actuator, sensor, or component.
- **Fault isolation:** determines which actuator, sensor, or component is faulty.

In other words, fault detection basically determines *when* a fault occurs (the *time* of fault occurrence), while the fault isolation determines *where* the fault is (the *location* of the fault).

1.3 FDI Methodologies

Fault Detection and Isolation (FDI) approaches can be divided into two categories: *hardware/physical redundancy* and *analytical redundancy* as shown in Figure 1.2.

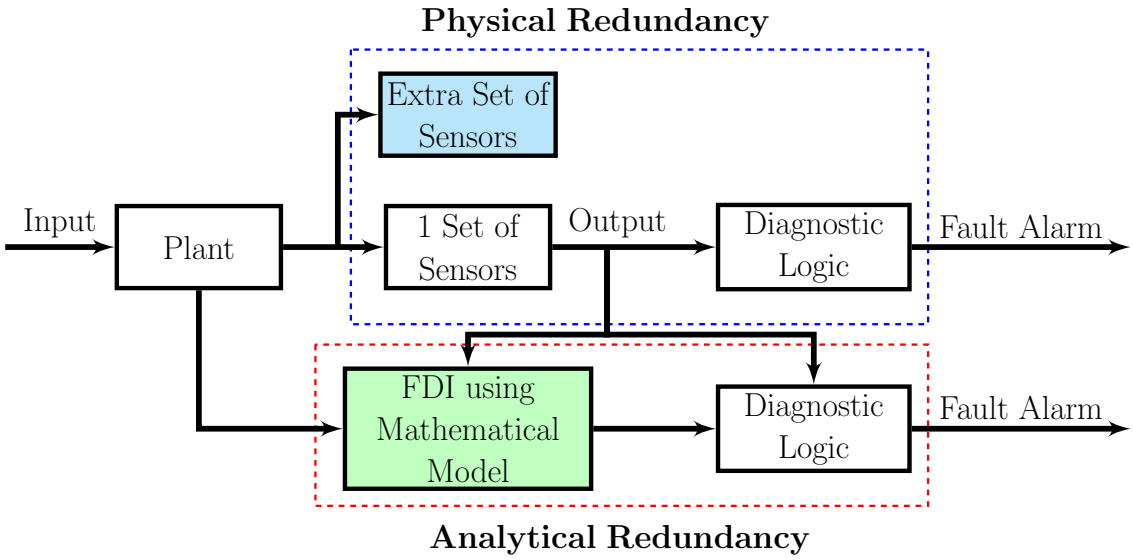


Figure 1.2: Physical redundancy versus analytical redundancy (adapted from [4]).

1.3.1 Hardware/Physical Redundancy

Physical redundancy is a popular option for applications that require high reliability and safety such as in large-scale WTs, flight-critical systems, and nuclear plants. In physical redundancy approaches, multiple sensors, actuators, or components are exploited at critical locations. In terms of sensor redundancy, multiple sensors measuring the same quantity are available, and any discrepancies between sensor measurements may indicate a fault within the monitored system.

A conventional fault diagnosis technique that is broadly used in industry is N-Modular Redundancy (NMR) approaches, where N is the number of modules available to perform a similar task. Dual Modular Redundancy (DMR) uses an extra set of equivalent modules, and in case of any discrepancy between the two outputs, it would be difficult to tell which one is faulty. When a DMR is available, a common FDI approach is first to detect the fault (or failure) by a direct comparison of the dual sensors, and then to perform isolation or identification tasks using one of analytical redundancy approaches (explained in Section 1.3.2) [5]. However, *the proposed FDI scheme in Chapter 5 performs both fault detection as well as fault isolation tasks.*

The most common form of NMR is the Triple Modular Redundancy (TMR) ($N = 3$) that is usually integrated with a majority voting mechanism [6]. In case of hardware redundancy, a Failure Mode and Effect Analysis (FMEA) is usually conducted to verify any defective element. However, the literature based on FDI from control system perspective do not usually include methodologies based on FMEA [7].

1.3.2 Analytical Redundancy

The analytical redundancy FDI methodologies can generally be categorized as follows [8–10]:

1. **Knowledge-Based Approaches:** These methods usually require an advance knowledge of the observed system such as training and learning data. Initially,

these methods employ a large amount of historic data to extract the underlying knowledge of the system variables. From this perspective, knowledge-based fault diagnosis is also known as data-driven fault diagnosis. Based on the extraction process, the knowledge-based approaches can be divided into the following categories [10]:

a. **Qualitative Knowledge-Based Methods:** Expert reasoning such as IF-THEN logic, or in general, rule-based fault diagnosis methods are among the most popular techniques in this category.

b. **Quantitative Knowledge-Based Methods:** These methods can be divided into the *statistical-analysis based methods* such as Principal Component Analysis (PCA), Support Vector Machine (SVM), or *non-statistical-analysis based methods* in which Neural Networks (NNs) have been the most known approach. In addition to the statistic and non-statistic fault diagnosis, a joint data-driven approach can also be used.

   Moreover, most Artificial Immune System (AIS) algorithms (including the ones used in this thesis) are considered to be in this category. It should be mentioned that *the proposed FDI scheme in Chapter 4 is a quantitative knowledge-based approach.*

2. **Signal-Based Approaches:** These methods are based on the measured signals

and symptom (pattern) analysis rather than the explicit input-output models. Signal-based approaches can be divided into three categories as follows: a) *Time–Domain Methods*, b) *Frequency-Domain Methods*, and c) *Time-Frequency Methods* [9].

3. **Model-Based Approaches:** These methods are usually divided into three categories as follows:

    a. **Observer-Based Approach:** This approach is the most common model-based FDI method. In observer-based FDI approach, system outputs are estimated from the measurements (or a subset of measurements) using either Luenberger observer in deterministic settings or Kalman filter in stochastic setting, and output estimation errors are then taken as residuals [11].

    b. **Parameter-Estimation (System Identification) Approach:** In this method, physical parameters of the system (such as mass, viscosity, or resistance) are estimated and then compared with the actual value of the parameter in faultless condition. Consequently, a fault can be detected if there is any discrepancy between the estimated and actual values.

    c. **Parity Relation Approach:** In this approach, a consistency check based on parity relations using inputs and output measurements are made. Consequently, faults can be detected if there is any inconsistency in the parity

relations.

4. **Hybrid Approaches:** These methods fuse different approaches mentioned above to improve the FDI performance.

1.4 Literature Review

1.4.1 Literature Review on Fault Diagnosis of Wind Turbines

Recent years have witnessed immense interest on fault diagnosis of WT systems. Odgaard *et al.* [12, 13] introduced a benchmark model of a WT at the system level for FDI and Fault Tolerant Control (FTC) applications, subject to sensor, actuator, and system faults. A review of the works that use mostly this benchmark model is provided next.

In [14], a model-based fault diagnosis approach is proposed by using interval-based Analytical Redundancy Relations (ARRs) and interval observers. An FDI and FTC approach based on Set-Valued Observers (SVOs) theory is developed in [15]. Chen *et al.* [16] have proposed a combined observer and Kalman filter approach in which the generalized likelihood ratio test and the cumulative variance index are used for residual evaluation. A diagnostic approach is proposed in [17] that uses a secondary H_∞ filtering scheme for FDI of a horizontal axis WT.

The work [18] addressed both the FDI as well as the FTC problem of the WT benchmark model through designing virtual actuators/sensors by using an input-output

model. In [18], the problem of the FDI was addressed in which interval observers are employed for fault detection and a row-reasoning technique is employed for the fault isolation stage. Moreover, a batch least square approach was used for fault estimation purposes.

The paper [19] proposed an estimation-based approach to perform the FDI of the WT benchmark model. In [19], fault detection task is achieved by a fault detection estimator, and a bank of fault isolation estimators are utilized to perform the isolation task. The work [20, 21] employed a set-membership FDI mechanism for the WT benchmark model.

In [22], a robust data-driven fault detection strategy that uses parity-space method is introduced, in which robust residual generators are constructed directly from the process measurements. The work [23] proposed a novel data-driven FDI scheme for a WT benchmark based on time-series and data analysis. In [23], the fault detection task is achieved by using the Gibbs Sampling algorithm, and the fault isolation task was performed by a Fuzzy/Bayesian network. The paper [24] utilized a mixed Bayesian/Set-membership approach for the fault detection and isolation of the WT. In [24], the fault detection task is formulated using the set-membership context; however, the fault isolation task is achieved based on the theoretical fault signature matrix, in which a novel Bayesian fault isolation methodology is proposed.

An FDI scheme based on the Takagi-Sugeno fuzzy models that are identified from

input-output measurements is proposed in [25–27]. Another data-driven approach that is successfully applied to FDI of WTs is the SVM in [28–30]. In [29], while most faults were detected and isolated through SVM, however, Kalman-like observers were designed to detect and isolate faults in the pitch actuator system. The work [30] also used a hybrid method for the FDI of WTs. Besides the SVM, a residual-based technique is used for the FDI of certain faults. More precisely, a three-layer Artificial Neural Network (ANN) was used to estimate the pitch angle for residual generation purposes.

A frequency-based fault detection technique is suggested in [31] for detecting changes in WT gearbox resonance frequencies. In [32], a fault diagnosis and FTC scheme is developed for the Individual Pitch Control (IPC) of WTs. The developed methodology in [32], activates and deactivates the IPC component of the WT blades based on the fault detection results.

Several FDI schemes applied to the wind turbine benchmark model are evaluated and compared in [13, 33, 34].

1.4.2 Literature Review on FDI using AIS Methodologies

The vertebrate Immune System (IS) (such as human IS) is an effective defensive mechanism that protects the body against any foreign invader. AISs model the natural IS by replicating and developing interesting paradigms that can be used in real-world applications. AIS is a sub-field of bio-inspired computing technologies that belongs to the

broader field of CI as shown in Figure 1.3.

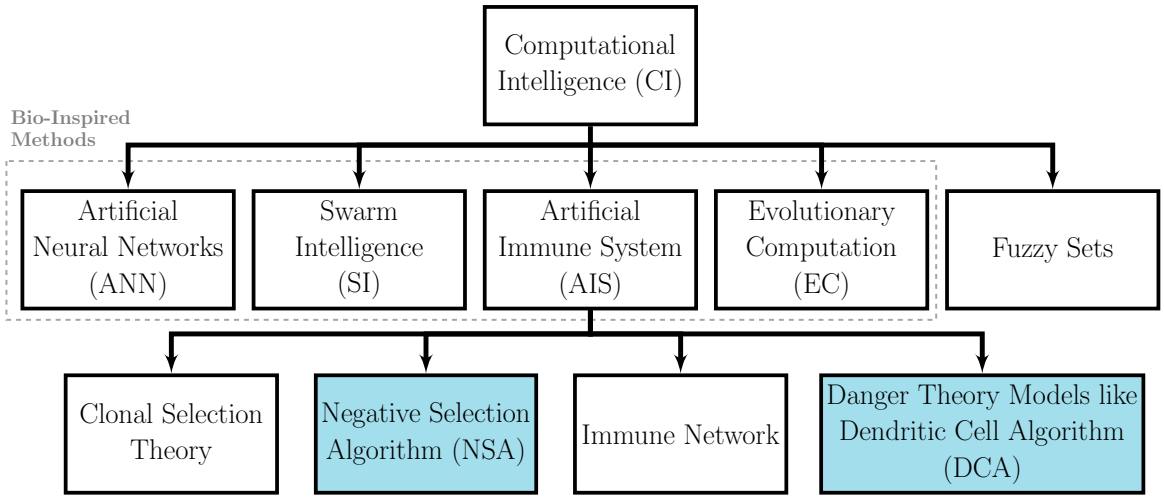


Figure 1.3: Branches of computational intelligence (CI).

1.4.2.1 Literature Review on Negative Selection Algorithm (NSA)

Negative Selection Algorithm (NSA) is used in various applications like: change detection, fault detection, pattern recognition, and particularly in computer security and network intrusion detection. Few works that utilized the NSA are mentioned below.

The Negative Selection Algorithm (NSA) was first introduced in 1994 by Forrest *et al.*, [35], as a *change detection* method in the field of computer security problem. In [35], the computer security problem was transformed to the problem of distinguishing self from non-self.

After the original work [35], there have been numerous attempts to utilize the NSA for computer security problems. In [36], both real-valued negative selection with fixed-sized detectors (RNSA) and variable-sized detectors (V-detector) are compared

with SVM and K-Nearest Neighbor (KNN), in which V-detector showed superior performances compared to the other ones. In [37], an intrusion detection mechanism is introduced based on the real-valued dual Negative Selection (NS), in which each newly generated detector has to go through three steps that are: 1) the detector should not be detected by the existing ones, 2) the detector should be discarded if it detects the self space, and 3) the distribution of detectors should be optimized with the goal of the detection efficiency. It should be noted that the first two steps of detector generation is similar to the original NSA. The papers [38, 39] also employed the NSA for network intrusion detection systems (IDS).

The work [40] integrated a hardware implementation of NSA with an embedded system for malware detection tasks. An adaptive NSA, in which the detector set is updated regularly, was proposed in [41] with the task of detecting any DoS (Denial of Service) attack, that is very important in cloud environments. The work [42] applied an improved version of the NSA to Wireless Sensor Networks (WSNs) in order to detect any anomaly. In [43], the NSA is utilized along with the artificial bee colony algorithm for intrusion detection in mobile ad hoc networks (MANETs).

The paper [44] worked on developing an intrusion detection model based on the NS mechanism. In [44], the problem of "black hole" in the original NSA was investigated. Consequently, the Minkowski distance is utilized as the matching rule in order to enhance the performance of the NSA by reducing the number of black holes resulting in a better

detection rate.

In [45,46], a motor bearing fault detection and diagnosis based on NSA is proposed. The work [47] investigated the use of real-valued NSA for aircraft fault detection, in which datasets from the NASA Ames man-in-the-loop high-fidelity C-17 flight simulator were employed in order to diagnose the behavior pattern of the aircraft flight. In [48], a bank of NSAs was developed to detect and isolate faults in a wind turbine system.

There were also few attempts to use NSA in function optimization problems. For instance, the paper [49] proposed a parallel implementation of the NSA to handle constraints in genetic algorithms. There have also been many attempts to address drawbacks of the original NSA as well as to improve the performance of the NSA. Consequently, various NSA versions have been developed, some of which are mentioned below:

A Distribution Estimation based NSA (*DENSA*) is proposed in [50] in which a Gaussian Mixture model is employed to fit the normal space (or self-space). The paper [51] proposed an Immune Optimization based Real-valued NSA (*IO-RNSA*) that has better time efficiency as well as less number of generated detector compared to traditional real-valued NSA. A boundary-aware NSA is developed in [52], in which the proposed algorithm interprets the training data as a group, unlike the original NSA that is an instance-based algorithm.

In [53] and [54], two novel NSAs with constant detectors were proposed to perform anomaly detection tasks. These novel versions of NSA are as follows: a Boundary-Fixed

NSA (*FB-NSA*) and a Fine Boundary-Fixed NSA (*FFB-NSA*). In both FB-NSA and FFB-NSA, a layer of detectors is generated to enclose the self-space.

The work [55, 56] applied the NSA along with the particle swarm optimization (PSO), called *NSA-PSO*, to an email detection system, in order to distinguish spam emails from non-spam ones. In [57], a modified NSA was developed by integrating a local selection differential evolution (DE) (hence, referred to as *NSA-DE*) for detector generation.

In [58], an intelligent fault diagnosis based on NSA was developed and successfully applied to fault diagnosis of large machine units in petrochemical industries. The paper [59] investigated the application of NSA in detecting faults in refrigeration systems. In [59], a differential encoding mechanism is utilized to reduce the size of the shape space. In addition, two matching rules, namely r-contiguous bits as well as Euclidean distance, are compared.

Tao *et al.* [60] proposed a Novel NSA (*NNSA*) in which a co-stimulation signal is required to start the detectors. NNSA utilizes the entropy information to estimate the detector coverage by evaluating the density of detectors. The paper [61] proposed a Modified NSA (*MNSA*) for diagnosis of voltage disturbances in distribution electrical systems. In [61], the affinity of the detector candidates are evaluated by calculating the value of population combination (VPC), which is used as a metric to eliminate the unnecessary detectors in the censoring phase. The work [62] used a popular version

of the NSA named V-detector for medical diagnosis purposes such as BUPSA Liver Disorder, Biomedical, and Breast Canver Wisconsin experimental results.

The paper [63] proposed a Bidirectional Inhibition Optimization R-variable NSA (referred to as *BIORV-NSA* or *BIOR-NSA*) to overcome one of the biggest problem of the original real-valued NSA that is the "black holes" issue, resulting in excessive, unnecessary detector generation to cover a small portion of the self-space.

A brief description of various NSA versions is available in Table 1.1.

Table 1.1: A summary of various NSA versions

Algorithm name	**Purpose of Development**	**Experimental Setup**
NSA-PSO [55, 56] (PSO: *P*article *S*warm *O*ptimization)	• Improving the random detector generation using PSO • Use of the Local Outlier Factor (LOF) as a fitness function	Email spam detection system
DENSA [50] *D*istribution *E*stimation based *NSA*	Efficient detector generation & more flexible boundary for self-space by using a Gaussian mixture model	Tested on NSL-KDD dataset
IO-RNSA [51] *I*mmune-*O*ptimization *R*eal-valued *NSA*	• Addressing the high time-complexity & redundant detector coverage by introducing an immune optimization mechanism	Tested on Breast Cancer Wisconsin, Iris, and Abalone UCI datasets
ENSA [64] *E*xtension *NSA*	• Efficient detector generation & better detector coverage • Improving the algorithm robustness	Tested on Iris dataset
FB-NSA & FFB-NSA [53, 54] (*B*oundary-*F*ixed *NSA*) (*F*ine *B*oundary-*F*ixed *NSA*)	• Non-random detector generation to reduce the training time • Generating a layer of detectors around the self-space	• Tested on the Fisher's Iris Data • Tested on ball bearing fault data
NSA-DE [57] (DE: *D*ifferential *E*volution)	• Use of DE for the detector generation phase • Overcoming the problem of overlapping detectors	Email spam detection system
NNSA (*N*ovel *NSA*) [60]	• Use of co-stimulation signal for activating the detector • Use of information entropy for a better detector coverage • New matching rule function for antigen recognition	• Used for recognition problems • Tested on Breast Cancer, Iris, and KDD CUP 99 UCI datasets
MNSA (*M*odified *NSA*) [61]	Changes in the censoring/learning process by using the Value of Population Combination (VPC) to evaluate the similarity of detector candidates	Diagnosis of voltage disturbances & high-impedance faults in distribution electrical systems
BIORV-NSA / BIOR-NSA [63] (*B*idirectional *I*nhibition *O*ptimization *R*-*V*ariable *NSA*)	• Overcoming the "black hole" issue • Removing unnecessary detectors	Tested on UCI datasets

1.4.2.2 Literature Review on Dendritic Cell Algorithm (DCA)

The Dendritic Cell Algorithm (DCA) was first proposed by Greensmith [65] as a part of the Danger project [66] with the task of applying a paradigm of state-of-the-art immunology to the computer network intrusion detection problems. The proposed DCA in [65] (referred to as the standard/classical/original version of DCA) includes many stochastic elements and has over ten parameters . Consequently, a deterministic version of the Dendritic Cell Algorithm (referred to as deterministic DCA (dDCA)) was proposed in [67] with fewer parameters, making it easier to analyze as well as apply.

Although the DCA has been mostly applied to computer security and intrusion detection problems, however, DCA has been successfully applied to other fields such as anomaly detection, fault detection, classification problems, pattern recognition, and *etc.* Next, few applications of DCA in different fields are provided.

The paper [68] proposed an effective spam filtering model in which the decisions of Naïve Bayes (NB) and SVM were fused and integrated with the DCA. Then, the proposed model was tested on two SMS spam datasets and showed promising results. In [69], the DCA was utilized for malware detection in Android smartphones.

The paper [70] proposed a rotating machinery fault diagnosis scheme based on the DCA, and also applied the developed methodology experimentally on a "Multi-Fault diagnosis rotating machinery test bed" showing its effectiveness.

In [71], a novel Fuzzy deterministic DCA (*FdDCA*) is proposed by combining

dDCA, K-means clustering, and fuzzy sets; and experimental results based on real-world datasets showed its applicability. In [72], a fault detection system was proposed to detect parametric faults in analog circuits using a fuzzy DCA.

The DCA has been also utilized in pattern recognition problems. For instance, the work [73] developed a plant classification method based on recognition of leaves. In [73], a wavelet transform was used to generate the texture features (features space) required by the classifier, in this case the DCA.

A modified DCA (*mDCA*) was introduced in [74] with the aim of an online error detection in robotic systems. The proposed mDCA was implemented successfully in both simulated robotic units as well as on a resource constrained micro-controller. The paper [75] presented the Diagnostic DCA (*D-DCA*) to perform online fault diagnosis in a robotic system, in case of stuck-at-faults conditions. In [76], dDCA was utilized for multi-path detection in a GPS time-series. The work [76] was implemented successfully on a static antenna, showing its capability to be used in more complicated GPS multipath detection problems. The paper [77] employed the DCA as a robot classifier and successfully implemented the DCA on a real robot.

Due to importance of data preprocessing phase in the DCA, many works were conducted to investigate the DCA data preprocessing phase that includes feature extraction as well as signal categorization steps. In [78], a statistical method, namely

principal component analysis (PCA), was utilized to automatically generate new features (by performing data dimension reduction) and then feeding them to the DCA.

Chelly and Elouedi [79] have investigated the use of Rough Set Theory (RST) and fuzzy rough set as a preprocessing step in the DCA, resulting in many versions of the DCA as follows:

- *RST-DCA* [80] that is based on the RST,
- *RC-DCA* [81] that is based on two main concepts of the RST, namely the REDUCT and the CORE.
- *QR-DCA* [82] that is based on a framework of the RST, namely the QuickReduct algorithm.
- *FBR-DCA* [83] that is based on the fuzzy RST and utilizes the Fuzzy Boundary Region (FBR) to ensure a better data preprocessing phase.
- *FRST-DCA* [84] that is also based on the Fuzzy RST.
- *RST-MFDCM* [85] that is based on a two-level hybrid model, in which the RST was utilized to perform the data pre-processing, and a Modified Fuzzy Dendritic Cell Method (MFDCM) (proposed in an earlier work [86]) was utilized to smooth the crisp separation among the DCs' context.
- *FLA-DCA* [87] that is an improved version of the FRST-DCA. Unlike FRST-DCA that uses fuzzy sets and membership functions, FLA-DCA uses a fuzzy lower approximation and similarity relations resulting in less calculations compared to

FRST-DCA.

Recently, researchers are working on FDI of aircraft by using an information processing algorithm that mimics the way dendritic cells operate in the IS. In their work [88, 89], *no DCA-based algorithm* (or any of its variants) is utilized. Instead, their work is based on generation of a self/non-self space (or equivalently, generation of non-self detectors) that requires a *training phase*. Therefore, the inputs to their FDI framework are sets of non-self detectors that are generated through NS mechanism. Their proposed framework would also not benefit from the main aspect of the DCA, that is the unsupervised learning perspective as well as the need for a training phase.

Table 1.2 lists few other popular variants of the DCA that had been successfully applied in real-world applications.

Table 1.2: Survey of various DCA versions

Algorithm name	**Purpose of Development**	**Experimental Setup**
dDCA [67] (*deterministic DCA*)	• Deterministic version of the original DCA • Fewer algorithm parameters • Simpler algorithm analysis	Tested on an Intrusion Detection System dataset, namely the ping scan data
mDCA [74] (*modified DCA*)	Online implementation on a resource-constraint microcontroller	Error detection in robotic systems
D-DCA [75] (*Diagnostic DCA*)	Online fault diagnosis	Online fault (stuck-at-fault condition) diagnosis in a robot
FdDCA [71] (*Fuzzy deterministic DCA*)	Smoothen the sharp boundaries between signals by using fuzzy sets and K-means clustering	Tested on several UCI datasets like Wisconsin Breast Cancer
Integrating SVM & NB with DCA [90] (SVM: Support Vector Machine) (NB: Naïve Bayes)	• Generating input signals for the DCA by using the SVM and the NB • Improving the classification performance	Mobile SMS spam datasets

1.5 Thesis Contributions

- In **Chapter 4**, multiple NSAs are employed to detect and isolate faults in the WT system. For the detection task, an NSA that is trained under normal or healthy operation of the WT is used to detect any abnormality in the system. For the isolation task, several NSAs are trained within a hierarchical architecture to isolate various faulty scenarios, including presence of simultaneous faults. A moving window filter is also utilized to improve the overall performance of the NSAs and to remove the outliers for enhancing the reliability of the proposed NSA-based FDI scheme. Moreover, the proposed scheme is compared with another data-driven methodology, namely the Support Vector Machine (SVM) method to demonstrate the advantages and capabilities of the proposed NSA approach. Finally, a nonparametric statistical comparison test is conducted to compare the proposed NSA scheme with the SVM.

- In **Chapter 5**, a framework based on the DCA is proposed to detect and isolate sensor faults in a system. In order to perform the goal of FDI for complex systems, a bank of DCA filters is applied in a distributed manner. The proposed DCA-based FDI methodology is then compared with the proposed NSA-based scheme developed in Chapter 4 in order to demonstrate its effectiveness and advantages. Like in Chapter 4, a nonparametric statistical comparison test is conducted to compare both the DCA-based as well as NSA-based FDI schemes.

1.6 Thesis Layout

The structure of the thesis is as follows:

- **Chapter 1** includes the literature review on two main topics. First, recent works on the fault diagnosis of the wind turbines are introduced in Section 1.4.1. Then, a literature review on fault diagnosis using Artificial Immune System (AIS) with the focus on Negative Selection Algorithm (NSA) (Section 1.4.2.1) and Dendritic Cell Algorithm (DCA) (Section 1.4.2.2) are presented.

- **Chapter 2** provides the background information and the detailed model of the wind turbine that is used in this thesis. The description as well as the model of common fault scenarios in the WT system are also presented.

- **Chapter 3** provides the necessary information about the nature of the AIS and biological paradigms that can be concluded from vertebrate immune systems. The main focus would be on background information of the two AIS algorithms considered in this thesis that are NSA (Section 3.1) and DCA (Section 3.2).

- **Chapter 4** introduces the proposed Fault Detection and Isolation methodology by using the Negative Selection Algorithm. The simulation results of implementing the proposed FDI methodology on the WT is also presented and compared with another data-driven technique, namely Support Vector Machine (SVM).

- **Chapter 5** presents the proposed Fault Detection and Isolation methodology by using the Dendritic Cell Algorithm. The simulation results of implementing the proposed FDI methodology on the WT is also presented and compared with the NSA-based methodology proposed in Chapter 4.

- **Chapter 6** presents the conclusion as well as some possible extensions for future work.

Chapter 2

Overview of Wind Turbine System

In this chapter, a wind model as well as the Wind Turbine (WT) component models, including mathematical formulation of the WT subsystems, and the complete nonlinear state-space representation of the WT are presented. The operational regions of the WT and the corresponding control system strategies are also provided. Finally, a number of most commonly occurring fault scenarios in the WT system are introduced, and details corresponding to the considered faults and their models are presented.

2.1 Wind Model

In this section, a wind model including all its components that affect the wind speed is presented. The wind model is composed of four main parts, namely [13]:

a) the mean wind speed $v_{\mathrm{m}}(t)$,

b) a stochastic behavior of the wind $v_{\mathrm{s}}(t)$,

c) the wind shear (difference in wind speed by height) effect $v_{\mathrm{ws}}(t)$,

d) the tower shadow effect $v_{\mathrm{ts}}(t)$.

$$v_{\mathrm{w}}(t) = v_{\mathrm{m}}(t) + v_{\mathrm{s}}(t) + v_{\mathrm{ws}}(t) + v_{\mathrm{ts}}(t), \tag{2.1}$$

where $v_{\mathrm{w}}(t)$ denotes the overall wind speed. The mean wind speed captures the slow variations of the wind and is obtained by low pass filtering the wind data measurements. The fast wind speed variations, also known as wind turbulence, determine the stochastic behavior of the wind. Modeling the dynamical properties of the turbulence can be obtained based on two types of spectra, namely: von Karman and Kaimal, although Kaimal's spectrum has been shown to be a better fit to experimental data [91]. More detail on the Kaimal model as well as other models describing the stochastic behavior of the wind is available in [91]. Both wind shear and the tower shadow effects (namely, $v_{\mathrm{ws}}(t)$ and $v_{\mathrm{ts}}(t)$, respectively) depend on the angular position of each rotor blade, and correspondingly, they are different from one blade to another. The equation describing the overall influence of the wind shear is given by [92]

$$\begin{aligned} v_{\mathrm{ws}}(t) = \frac{2}{3} v_{\mathrm{m}}(t) \sum_{b=1}^{3} \Bigg[& \frac{\alpha}{3} \left(\frac{R}{H} \right) \cos \theta_b(t) + \frac{\alpha(\alpha-1)}{8} \left(\frac{R}{H} \right)^2 \cos^2 \theta_b(t) \\ & + \frac{\alpha(\alpha-1)(\alpha-2)}{30} \left(\frac{R}{H} \right)^3 \cos^3 \theta_b(t) \Bigg], \end{aligned} \tag{2.2}$$

where R denotes the radius of the rotor, H denotes the tower height, α denotes the empirical wind shear component, and $\theta_b(t)$, $b = 1, 2, 3$ denotes the angular position of

each blade (also known as the azimuthal angle) with respect to an arbitrary position, $\theta_r(t)$. Correspondingly, one has $\theta_1(t) = \theta_r(t)$, $\theta_2(t) = \theta_r(t) + (2/3)\pi$, and $\theta_3(t) = \theta_r(t) + (4/3)\pi$.

The total tower shadow effect is modeled according to [92]

$$v_{\text{ts}}(t) = \frac{2m}{3r^2} \sum_{b=1}^{3} \bar{\theta}_b(t) \left(\psi(t) + \upsilon(t)\right), \tag{2.3}$$

where

$$\begin{aligned}
\psi(t) &= a^2 \frac{R^2 - r_0^2}{(R^2 + r_0^2)\sin(\bar{\theta}_b(t))^2 + k^2)}, \\
\upsilon(t) &= a^2 k^2 \frac{(r_0^2 - R^2)(r_0^2 \sin(\bar{\theta}_b(t))^2 + k^2)}{R^2 \sin(\bar{\theta}_b(t))^2 + k^2}, \\
m &= 1 + \frac{\alpha(\alpha - 1) r_0^2}{8H^2}, \\
\bar{\theta}_b(t) &= \theta_{\text{r}}(t) + \frac{2\pi(b-1)}{3} - \text{floor}\left(\frac{\theta_{\text{r}}(t) + \frac{2\pi(b-1)}{3}}{2\pi}\right) 2\pi,
\end{aligned}$$

a denotes the radius of the tower, r_0 denotes the radius of the blade hub, and k denotes the distance between the tower midline and the blade. Figure 2.1 illustrates the wind model parameters used in equation (2.3). The floor function, $\text{floor}(x) = \lfloor x \rfloor$, yields the largest integer less than or equal to x.

Figure 2.2 illustrates the wind speed sequence that is collected from actual measured wind data. This predefined wind speed sequence is used in the WT benchmark model [13].

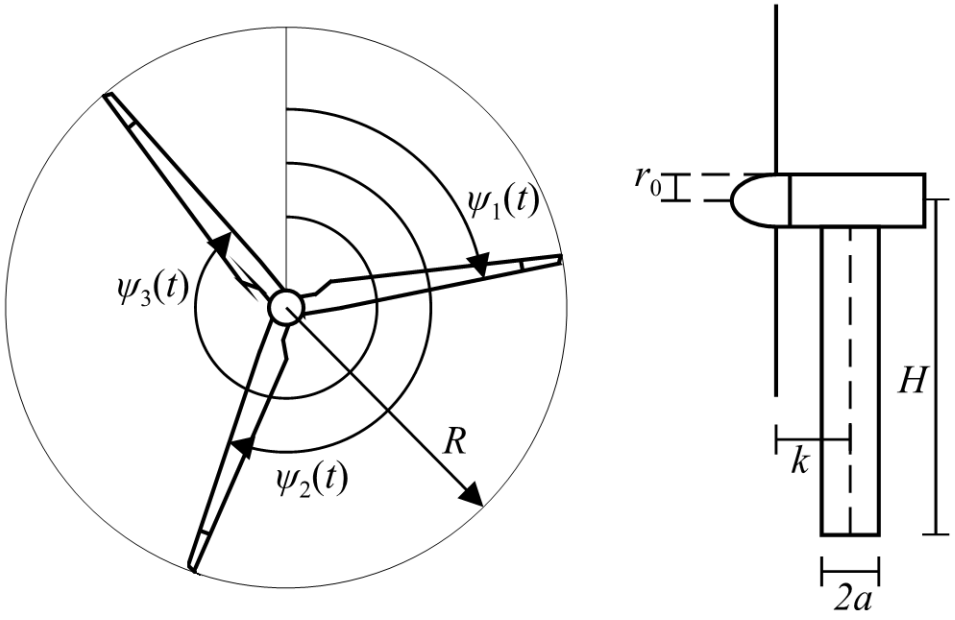


Figure 2.1: Overview of the wind turbine subsystems (adapted from [93]).

2.2 Wind Turbine Model

In the WT model selected, the turbine has a given power capacity that is equipped with three-bladed horizontal axis turbine, a full converter coupling and a generator that is both variable speed and pitch-controlled. An insight to a typical modern high-power WT showing its main components is illustrated in Figure 2.3.

The WT model consists of three main subsystems, namely: blade and pitch subsystem (combining the aerodynamic and the pitch actuator model), drive-train subsystem, and the generator and converter subsystem. The interconnections among the wind turbine components and modules are depicted in Figure 2.4. The rotor and generator variables are designated by the subscripts r and g, respectively.

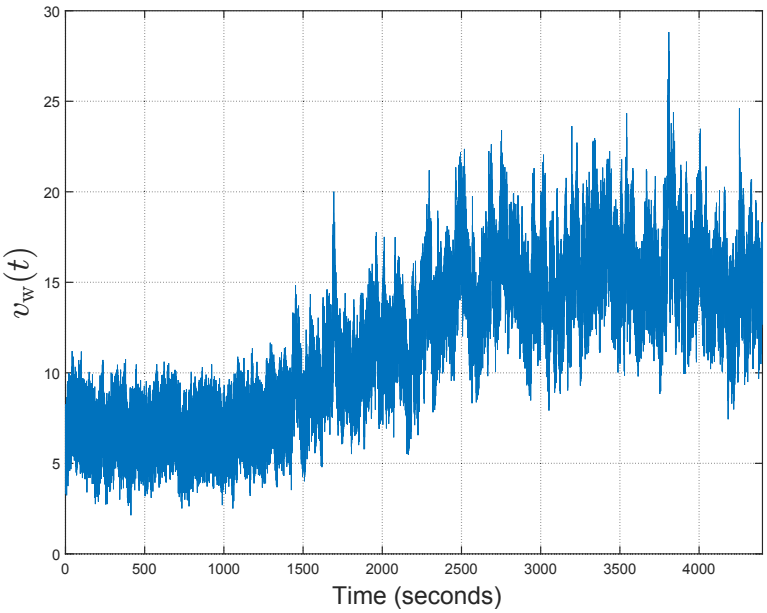


Figure 2.2: Illustration of the wind speed, $v_w(t)$, used in the WT benchmark model.

2.2.1 Aerodynamics Model

In this section, the basic mechanism of WT and how the wind energy is converted to the rotational motion of the WT rotor blades are provided. A significant parameter in the aerodynamics performance of the WT is the power coefficient, C_p, which is defined as the ratio between the captured power by the rotor $P_r(t)$ and the total wind power $P_{\text{wind}}(t) = \frac{1}{2}\rho A v_w^3(t)$, where ρ denotes the air density and A denotes the rotor swept area. Therefore, the power delivered to the rotor is given by

$$P_r(t) = P_{\text{wind}}(t) C_p(\lambda(t), \beta(t)). \tag{2.4}$$

As noted in equation (2.4), C_p is a function of the pitch angle $\beta(t)$ and the tip-speed ratio $\lambda(t)$. The tip-speed ratio is the ratio between the tip speed of the blade ($v_{\text{tip}}(t)$)

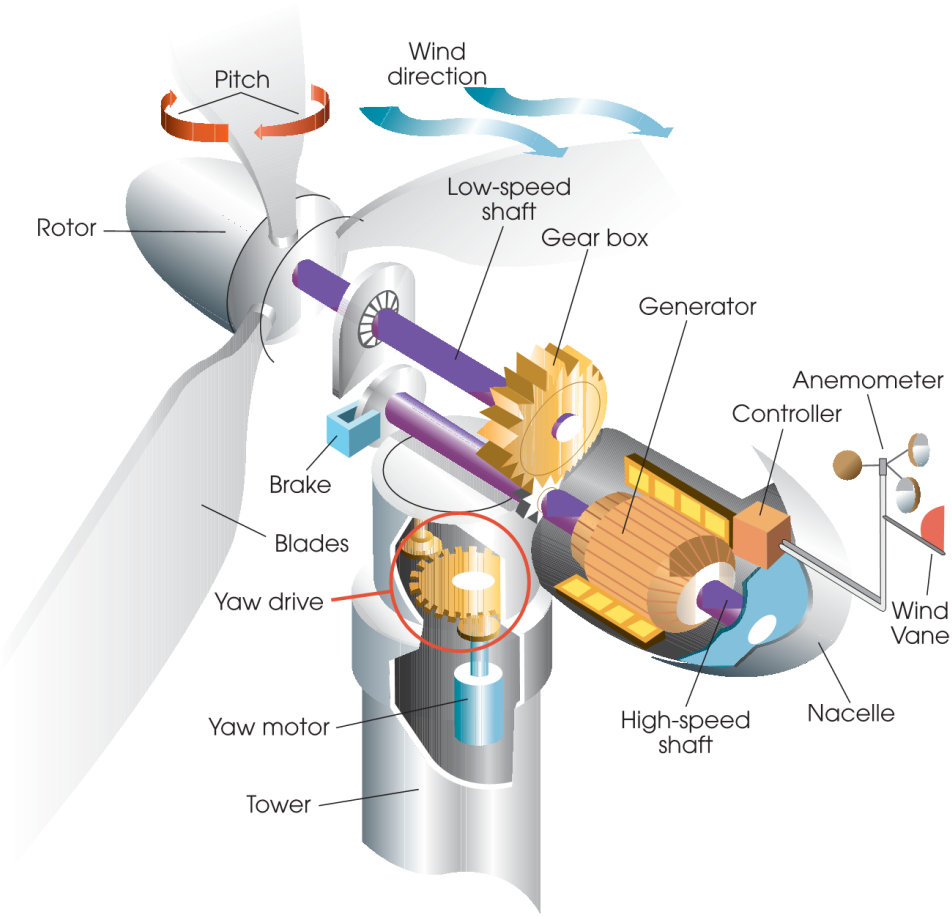


Figure 2.3: Main Components of a typical modern high-power WT (reprinted from [94]).

and the wind velocity ($v_{\mathrm{w}}(t)$), that is $\lambda(t) = \frac{v_{\mathrm{tip}}(t)}{v_{\mathrm{w}}(t)} = \frac{\omega_r(t)R}{v_{\mathrm{w}}(t)}$, where $\omega_r(t)$ denotes the rotor rotational speed and R denotes the rotor radius (length of one blade). Typical variations of the power coefficient, C_p-coefficient, which is plotted based on the look-up table available in the WT benchmark model, is illustrated in Figure 2.5. $C_{p,\max}$ denotes the maximum power coefficient at point (λ_o, β_o). The angle β_o is usually very small ($\beta \approx 0$) meaning that the maximum power capture occurs at $\beta = 0$. On the other hand, maximum energy conversion efficiency occurs at λ_o. Due to this fact, fixed-speed

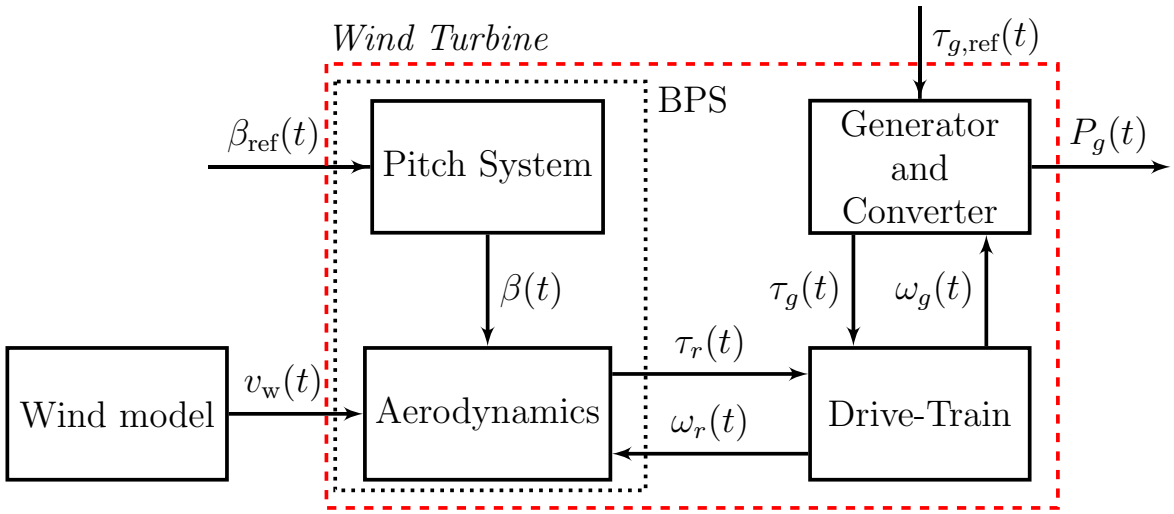


Figure 2.4: Overview of the wind turbine subsystems.

WTs ($\beta = 0$) operate with maximum efficiency only at a unique v_{w} [95].

The aerodynamics of the WT is expressed as the torque acting on the rotor blades. This rotor torque, τ_r, (also known as the aerodynamic torque) can now be expressed as follows:

$$\tau_r(t) = \frac{P_r(t)}{\omega_r(t)} = \frac{1}{2\omega_r(t)} \rho A v_{\mathrm{w}}^3(t) C_p(\lambda(t), \beta(t)). \tag{2.5}$$

2.2.2 Pitch Actuator System

The WT hydraulic pitch system consists of actuators that adjust the pitch angle, $\beta(t)$, of the rotor blade by rotating it. The pitch actuator dynamics can be modeled as a

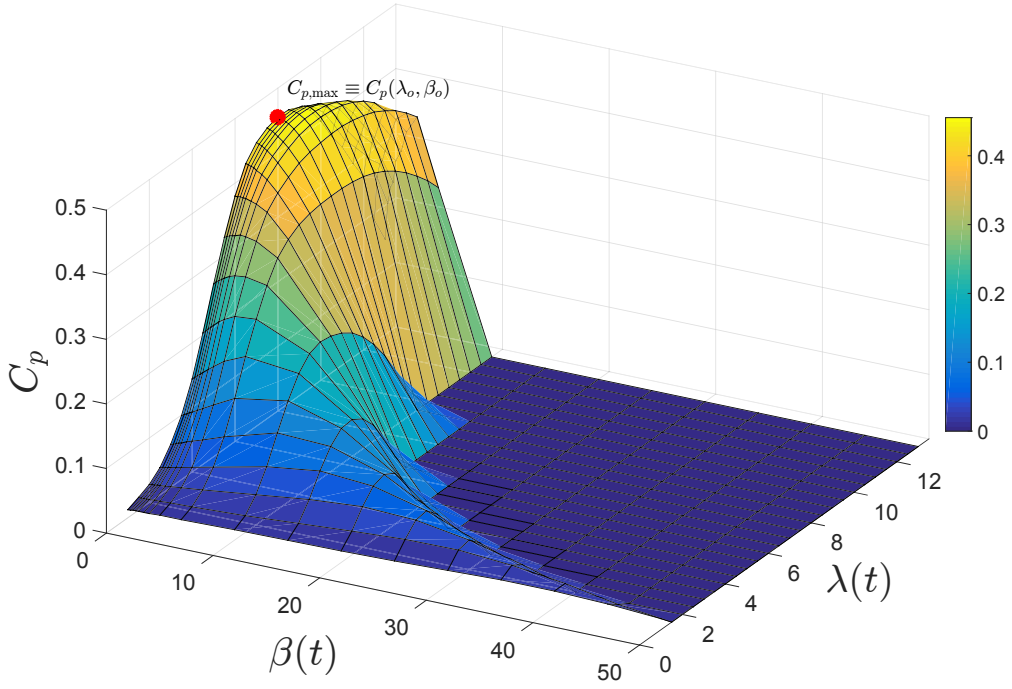


Figure 2.5: Typical variations of the power coefficient, C_p, for a variable-pitch WT. Note that negative values are set to zero.

second-order system as follows:

$$\begin{bmatrix} \dot{\beta}_i(t) \\ \ddot{\beta}_i(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \begin{bmatrix} \beta_i(t) \\ \dot{\beta}_i(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} \beta_{i,\,\mathrm{ref}}(t), \tag{2.6}$$

where $i = 1, 2, 3$ corresponds to the i-th blade, ζ and ω_n denote the damping ratio and the natural frequency of the pitch actuator model, and $\beta_{i,\,\mathrm{ref}}(t)$ denotes the reference pitch angle that is provided by the WT controller based on the operational region of the turbine. Figure 2.6 illustrates the pitch actuator model. As can be seen in Figure 2.6, there are constraints on both the pitch slew rate as well as the pitch angle that are

embedded within the WT benchmark model. However, according to [95], a typical range for the slew rate is $\pm 10°/\text{s}$, and a typical range of pitch angle is $\beta \in [-2, 30]°$.

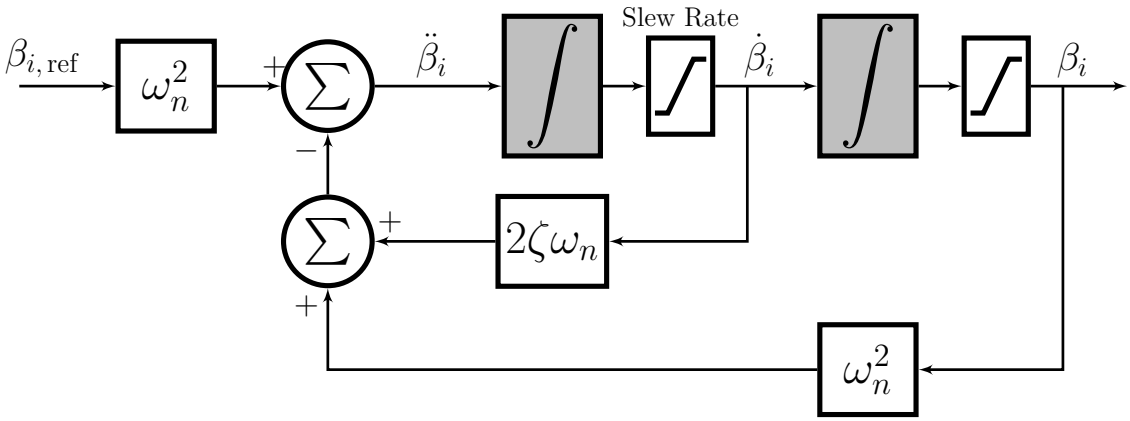


Figure 2.6: A schematic of the pitch actuator system for the i-th blade.

The pitch actuator system together with the aerodynamic model make up the Blade and Pitch System (BPS). Table 2.1 lists the parameters of the pitch actuator system and the aerodynamic model.

Table 2.1: Parameters of the pitch actuator and aerodynamic model (BPS).

ζ	ω_n	ρ	R
0.6	11.11 $\frac{\text{rad}}{\text{s}}$	1.225 $\frac{\text{kg}}{\text{m}^3}$	57.5 m

2.2.3 Drive-Train Model

The Drive-Train (DT) is a mechanical linkage connecting the low-speed shaft (rotor) to the high-speed shaft (generator). The DT includes a gearbox that transfers the rotor torque to the generator, and results in an increase of the rotor rotational speed to a

higher speed that is required by the generator. This process is illustrated in Figure 2.7.

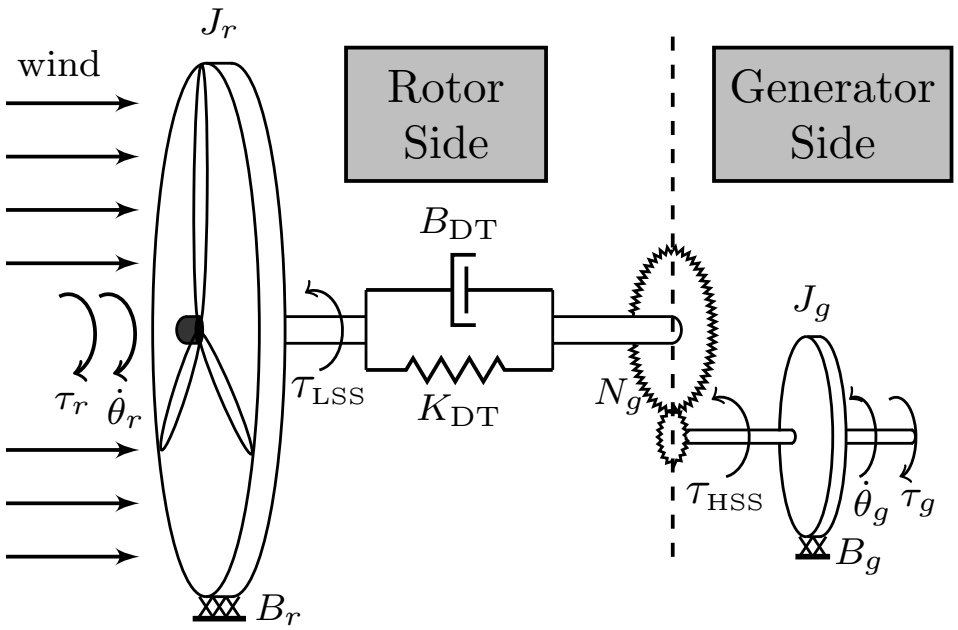


Figure 2.7: A schematic of the WT drive-train.

Applying the Newton's second law for rotation for both rotor as well as generator results in the following equations, namely

$$J_r\ddot{\theta}_r(t) = \tau_r(t) - \tau_{\mathrm{LSS}}(t) - B_r\omega_r(t), \tag{2.7}$$

$$J_g\ddot{\theta}_g(t) = \tau_{\mathrm{HSS}}(t) - \tau_g(t) - B_g\omega_g(t), \tag{2.8}$$

where $\theta_r(t)$ and $\theta_g(t)$ denote the rotational angles of the low-speed shaft (LSS) and high-speed shaft (HSS) respectively, $\omega_r(t)$ denotes the LSS (rotor) speed ($\omega_r(t) = \dot{\theta}_r(t)$), $\omega_g(t)$ denotes the HSS (generator) speed ($\omega_g(t) = \dot{\theta}_g(t)$), J_r and J_g denote the moments

of inertia of the low-speed and high-speed shafts, respectively, B_r and B_g denote the viscous frictions of the low-speed and high-speed shafts, respectively, and $\tau_g(t)$ denotes the generator torque (explained in Section 2.2.4). $\tau_{\mathrm{LSS}}(t)$ and $\tau_{\mathrm{HSS}}(t)$ denote the torques acting on the LSS and HSS, which can be obtained as follows

$$\tau_{\mathrm{LSS}}(t) = K_{\mathrm{DT}}\theta_{\Delta}(t) + B_{\mathrm{DT}}\dot{\theta}_{\Delta}(t), \tag{2.9}$$

$$\tau_{\mathrm{HSS}}(t) = \frac{\eta_{\mathrm{DT}}\,\tau_{\mathrm{LSS}}(t)}{N_g}, \tag{2.10}$$

where K_{DT} and B_{DT} denote the torsional stiffness and the torsional damping coefficient of the drive-train, respectively, η_{DT} denotes the efficiency of the drive-train, and $\theta_{\Delta}(t)$ denotes the torsional angle of the drive-train (describing the twist of the flexible shaft) that can be obtained using equation (2.11) as follows

$$\theta_{\Delta}(t) = \int_0^t \left(\omega_r(\alpha) - \frac{\omega_g(\alpha)}{N_g}\right) \mathrm{d}\alpha, \tag{2.11}$$

Note that the shaft angles $\theta_r(t)$ and $\theta_g(t)$ (except for $\theta_{\Delta}(t)$) are not of interest in modeling of the DT dynamics [93].

The differential equations describing the complete dynamics of the drive-train model can be obtained by differentiating equation (2.11) and also by combining equations (2.9) and (2.10) into equations (2.7) and (2.8) as follows

$$J_r\,\dot{\omega}_r(t) = \tau_r(t) - K_{\mathrm{DT}}\,\theta_{\Delta}(t) - (B_{\mathrm{DT}} + B_r)\,\omega_r(t) + \frac{B_{\mathrm{DT}}}{N_g}\,\omega_g(t), \tag{2.12}$$

$$J_g\,\dot{\omega}_g(t) = \frac{\eta_{\mathrm{DT}} K_{\mathrm{DT}}}{N_g}\,\theta_\Delta(t) + \frac{\eta_{\mathrm{DT}} B_{\mathrm{DT}}}{N_g}\,\omega_r(t) - \left(\frac{\eta_{\mathrm{DT}} B_{\mathrm{DT}}}{N_g^2} + B_g\right)\omega_g(t) - \tau_g(t), \quad (2.13)$$

$$\dot{\theta}_\Delta(t) = \omega_r(t) - \frac{1}{N_g}\,\omega_g(t). \quad (2.14)$$

The parameters of the DT model is available in Table 2.2. The differential equations (2.12) – (2.14) are rewritten in state-space format as follows

$$\begin{bmatrix} \dot{\omega}_r(t) \\ \dot{\omega}_g(t) \\ \dot{\theta}_\Delta(t) \end{bmatrix} = \mathbf{A}_{\mathrm{DT}} \begin{bmatrix} \omega_r(t) \\ \omega_g(t) \\ \theta_\Delta(t) \end{bmatrix} + \mathbf{B}_{\mathrm{DT}} \begin{bmatrix} \tau_r(t) \\ \\ \tau_g(t) \end{bmatrix}, \quad (2.15)$$

with $\mathbf{A}_{\mathrm{DT}}$ and $\mathbf{B}_{\mathrm{DT}}$ are given by

$$\mathbf{A}_{\mathrm{DT}} = \begin{bmatrix} -\frac{B_{\mathrm{DT}}+B_r}{J_r} & \frac{B_{\mathrm{DT}}}{N_g J_r} & -\frac{K_{\mathrm{DT}}}{J_r} \\ \frac{\eta_{\mathrm{DT}} B_{\mathrm{DT}}}{N_g J_g} & -\left(\frac{\eta_{\mathrm{DT}} B_{\mathrm{DT}}}{N_g^2 J_g} + \frac{B_g}{J_g}\right) & \frac{\eta_{\mathrm{DT}} K_{\mathrm{DT}}}{N_g J_g} \\ 1 & -\frac{1}{N_g} & 0 \end{bmatrix}, \; \mathbf{B}_{\mathrm{DT}} = \begin{bmatrix} \frac{1}{J_r} & 0 \\ 0 & \frac{1}{J_g} \\ 0 & 0 \end{bmatrix}.$$

Table 2.2: Drive-train parameters.

K_{DT}	J_r	J_g	N_g
$2.7 \times 10^9\ \frac{\mathrm{Nm}}{\mathrm{rad}}$	$55 \times 10^6\ \mathrm{kg \cdot m^2}$	$390\ \mathrm{kg \cdot m^2}$	95
B_{DT}	B_r	B_g	η_{DT}
$775.49\ \frac{\mathrm{Nm \cdot s}}{\mathrm{rad}}$	$7.11\ \frac{\mathrm{Nm \cdot s}}{\mathrm{rad}}$	$45.6\ \frac{\mathrm{Nm \cdot s}}{\mathrm{rad}}$	0.97

2.2.4 Generator and Converter Model

The most common generator that is used in the variable-speed WT is the doubly-fed induction generator [96]. Since the generator torque, τ_g cannot be changed instantaneously, the WT generator and converter dynamics at the system level, can be approximated by a first order dynamics as in [13] and [15], namely

$$\frac{\tau_g(s)}{\tau_{g,\mathrm{ref}}(s)} = \frac{1}{T_{gc}\, s + 1}, \tag{2.16}$$

where T_{gc} denotes the time constant of the first order system, and $\tau_{g,\mathrm{ref}}$ denotes the generator reference torque that is provided by the wind turbine controller according to the operational region of the WT as described subsequently in Section 2.3. The time-domain version of equation (2.16) is given as follows:

$$\dot{\tau}_g(t) = -\frac{1}{T_{gc}}\,\tau_g(t) + \frac{1}{T_{gc}}\,\tau_{g,\mathrm{ref}}(t). \tag{2.17}$$

Figure 2.8 shows the converter dynamics. Similar to the pitch system, there are bounds on both the slew rate, $\dot{\tau}_g$, as well as the generator torque, τ_g that are embedded in the WT benchmark model.

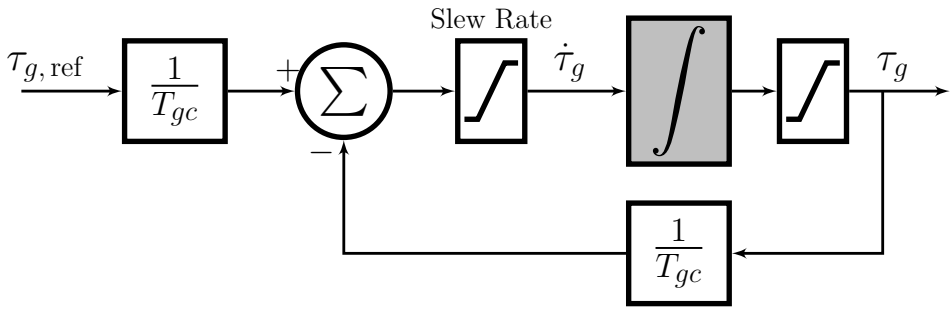


Figure 2.8: A schematic of the converter.

The power that is produced by the generator, P_g, is given by

$$P_g(t) = \eta_g \, \omega_g(t) \, \tau_g(t), \tag{2.18}$$

where η_g denotes the generator efficiency. The parameters of generator and converter subsystem used in this benchmark model is available in Table 2.3. More details on the generator and the converter model are available in [97].

Table 2.3: Parameters of the generator and converter subsystem.

Power Capacity	η_g	T_{gc}
4.8 MW	0.98	0.02 s

2.2.5 Combined Model

The complete model of the wind turbine can be obtained by combining the models developed in Sections 2.2.1 – 2.2.4. Figure 2.9 illustrates the interactions among all WT subsystems and provides the available sensor measurements in the benchmark model. A snapshot of the WT Simulink model (see Figure B.1) is included in Appendix B.

The following nonlinear state-space representation of the wind turbine system, which is affine in the control input, can be obtained by combining the equations (2.5),

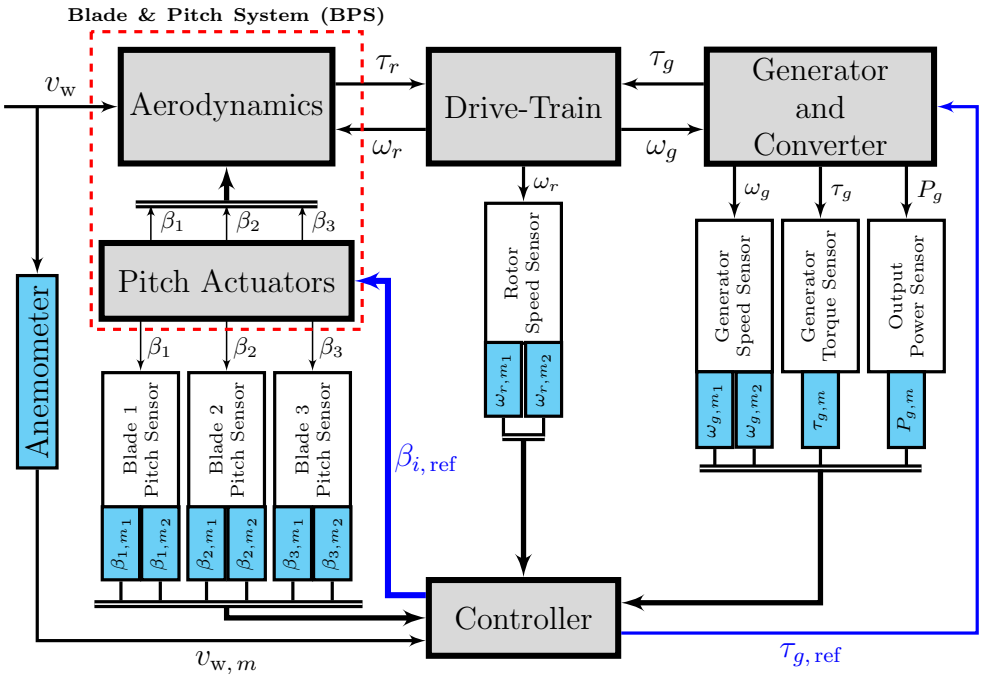


Figure 2.9: Schematic of WT subsystems.

(2.6), (2.15), and (2.17) as follows:

$$\dot{x}(t) = f(x(t), v_{\mathrm{w}}(t)) + \mathbf{G}\,u(t)$$

$$\begin{bmatrix} \dot{\omega}_r(t) \\ \dot{\omega}_g(t) \\ \dot{\theta}_\Delta(t) \\ \dot{\tau}_g(t) \\ \dot{\beta}_i(t) \\ \ddot{\beta}_i(t) \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{P_r(\omega_r(t),\beta_i(t),v_{\mathrm{w}}(t))}{J_r\,\omega_r(t)} - \frac{B_{\mathrm{DT}}+B_r}{J_r}\,\omega_r(t) + \frac{B_{\mathrm{DT}}}{J_r N_g}\,\omega_g(t) - \frac{K_{\mathrm{DT}}}{J_r}\,\theta_\Delta(t) \\ \frac{\eta_{\mathrm{DT}} B_{\mathrm{DT}}}{J_g N_g}\,\omega_r(t) - \left(\frac{\eta_{\mathrm{DT}} B_{\mathrm{DT}}}{J_g N_g^2} + \frac{B_g}{J_g}\right)\omega_g(t) + \frac{\eta_{\mathrm{DT}} K_{\mathrm{DT}}}{J_g N_g}\,\theta_\Delta(t) - \frac{1}{J_g}\tau_g(t) \\ \omega_r(t) - \frac{1}{N_g}\,\omega_g(t) \\ -\frac{1}{T_{gc}}\,\tau_g(t) \\ \dot{\beta}_i(t) \\ -\omega_n^2\,\beta_i(t) - 2\zeta\omega_n\,\dot{\beta}_i(t) \end{bmatrix}}_{\mathbf{A}} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{T_{gc}} & 0 \\ 0 & 0 \\ 0 & \omega_n^2 \end{bmatrix}}_{\mathbf{G}} u(t) \tag{2.19}$$

where the wind speed, $v_{\mathrm{w}}(t)$, represents the exogenous input, the state vector $x(t) \in \mathbb{R}^{10}$ and the controlled input $u(t) \in \mathbb{R}^{4}$ are given by

$$x(t) = [\omega_r(t) \quad \omega_g(t) \quad \theta_\Delta(t) \quad \tau_g(t) \quad \beta_1(t) \quad \dot{\beta}_1(t) \quad \beta_2(t) \quad \dot{\beta}_2(t) \quad \beta_3(t) \quad \dot{\beta}_3(t)]^{\mathrm{T}},$$

$$u(t) = [\tau_{g,\,\mathrm{ref}}(t) \quad \beta_{1,\,\mathrm{ref}}(t) \quad \beta_{2,\,\mathrm{ref}}(t) \quad \beta_{3,\,\mathrm{ref}}(t)]^{\mathrm{T}}.$$

It should be noted that all three pitch actuators receive the same pitch reference angles in this benchmark model. The output equation based on the available sensor measurements (refer to Figure 2.9) is expressed as follows:

$$y_m(t) = [\omega_{r,m_j}(t) \quad \omega_{g,m_j}(t) \quad \tau_{g,m}(t) \quad P_{g,m}(t) \quad \beta_{i,m_j}(t) \quad v_{\mathrm{w},m}(t)]^{\mathrm{T}} + \nu(t),$$

where $i \in \{1, 2, 3\}$ refers to the i-th blade, and $j \in \{1, 2\}$ corresponds to the redundant sensor measurements available for the particular WT parameter. In the WT benchmark model [13], it is assumed that redundant sensors *cannot* be faulty simultaneously, allowing the control system to reconfigure itself in case of occurrence of a sensor fault. The WT sensor measurements are perturbed with an additive noise, $\nu(t)$, to represent and capture the non-ideality of the sensor readings. The added noise signals are modeled as Gaussian random processes with means and variances (based on practical experiences [13]) as presented in Table 2.4. It should be noted that the sampling period selected for discretizing the continuous-time model (2.19) is set to 0.1 seconds.

Table 2.4: Measurement noise parameters.

	β_i	ω_r	ω_g	P_g	τ_g	v_{w}
Mean	0	0	0	0	0	$1.5\,\frac{\mathrm{m}}{\mathrm{s}}$
Variance	$0.2\,^\circ$	$0.025\,\frac{\mathrm{rad}}{\mathrm{s}}$	$0.05\,\frac{\mathrm{rad}}{\mathrm{s}}$	$1\times10^3\,\mathrm{W}$	$90\,\mathrm{Nm}$	$0.5\,\frac{\mathrm{m}}{\mathrm{s}}$

2.3 Operational Regions of the Wind Turbine

Variable-speed wind turbines have four main operational zones depending on the mean wind speed. These control zones are depicted in Figure 2.10 and are defined as follows:

- **Zone 1**: In zone 1, the WT is not operational and is waiting for higher wind speeds ($0-3\,\mathrm{m/s}$). The WT enters zone 2 when wind speed reaches the *cut-in speed* (the minimum speed at which the WT delivers useful power).

- **Zone 2**: This zone starts once the mean wind speed reaches the cut-in speed. Zone 2 is referred to as the partial-load region (or the power optimization region). The wind turbine is operating in zone 2 ($3-12.5\,\mathrm{m/s}$) with the goal of optimizing the generated power by maximizing the wind energy capture. This task can be achieved at the maximum power coefficient point $C_{p,\max}$ at which $\beta = 0^\circ$ (see Figure 2.5). Consequently, the controller sets $\beta_{i,\mathrm{ref}} = 0$ for an optimal power capture.

- **Zone 3**: In zone 3 ($12.5-25\,\mathrm{m/s}$), the WT is operating at its rated speed (the speed at which the rated power of the WT is produced), and the controller should maintain the power reference P_{ref} by limiting the captured wind energy so that

the mechanical load is not exceeded. P_{ref} can be obtained by maintaining the $\tau_{g,\text{ref}}$ in equation (2.18), so that $\tau_{g,\text{ref}} = \frac{P_{\text{ref}}}{\eta_g \omega_g}$. Commonly, a Proportional-Integral (PI) controller is used to maintain $\omega_g(t)$ at its nominal value by controlling $\beta_{i,\text{ref}}$. This zone is referred to as the full-load region.

- **Zone 4**: In zone 4, the wind turbine is shut down at *cut-out speed* (the maximum speed at which the WT is allowed to operate) due to high wind speeds ($> 25\,\text{m/s}$) to avoid any damage to the turbine [13].

Zones 1 and 4 are not considered in the benchmark model since the focus is on WT faults under normal operations, and consequently, only the control modes of zones 2 and 3 are taken into account in this work. The controller outputs under both modes are the reference pitch angle $\beta_{i,\text{ref}}$ and the generator reference torque $\tau_{g,\text{ref}}$. In the power optimization region (zone 2), the pitch reference is set to zero for optimal power capture from the wind. However, in the constant power generation region (zone 3), the controller outputs are produced in such a manner to maintain the nominal generator speed [13,98].

2.4 Common Faults in the Wind Turbine Systems

In this section, the commonly occurring faults on the WT components and their effects on the system behavior are presented. The details corresponding to the considered faults

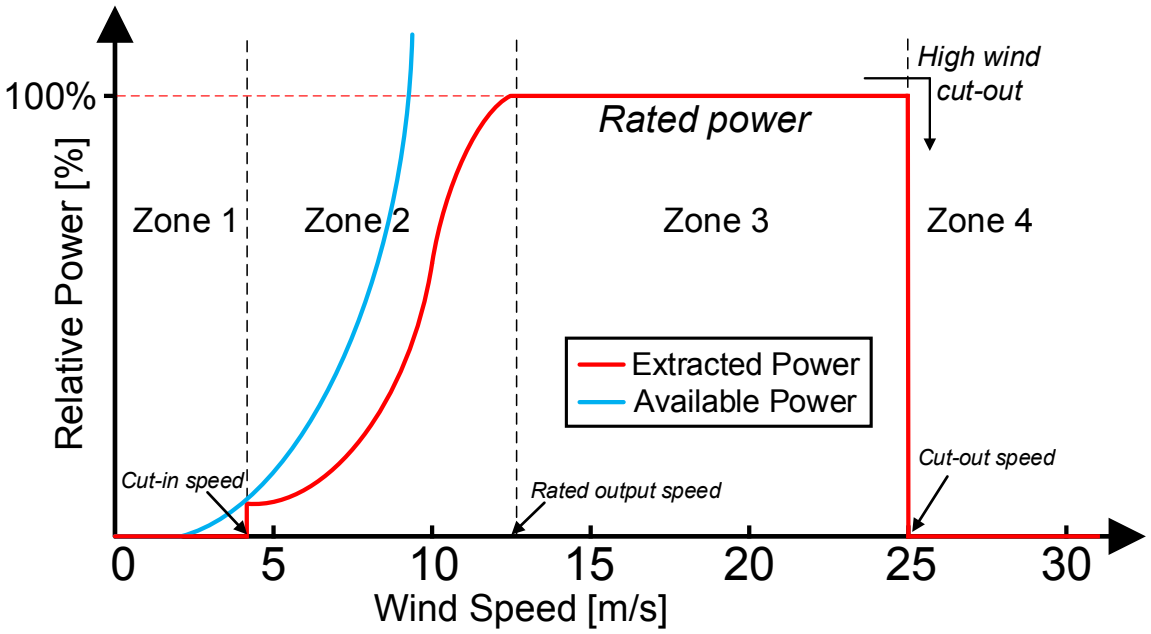


Figure 2.10: Reference power curve of the WT based on the wind speed.

in this thesis are also provided below.

Table 2.5 provides a list of the most significant faults in the WT (at the system level), including faults that are considered in the benchmark models [12, 13, 99].

Table 2.5: Critical faults that are considered in the WT benchmark model [13, 99].

Fault Category	**Fault**		**Type**
Sensor Faults	Pitch position measurement		Fixed value / Scaling / Bias
	Rotor speed measurement		Fixed value / Scaling
	Generator speed measurement		
	Generated electrical power		Scaling
	Tower top accelerometer		Offset
	Blade root bending moment sensor		Scaling
	Low speed shaft position encoder (Azimuth angle)		Bit error
Actuator Faults	Pitch actuator	High air content in the oil	Changed dynamics
		Hydraulic leakage	
		Pump wear	
	Generator torque		Offset / Changed dynamics
	Yaw drive		Stuck
System Faults	Drive-train	Wear and tear	Change in the drive-train damping
	Change in the WT aerodynamics	Accumulation of the debris on the blade	Scaling

2.4.1 Selected Fault Scenarios

The details corresponding to the considered faults in this work are also provided below. First, three specific and commonly occurring faults that are affecting the WT measured variables are considered as described below.

1. **Fault 1 (System Fault)**: This fault occurs due to accumulation of debris on the blade (such as dirt, *etc.*) and results in a change in the aerodynamics of the WT. Its occurrence directly affects the generated electrical power P_g (by usually lowering the obtained power). The corresponding fault model is given by

$$\tilde{P}_{g,m}(t) = \tilde{P}_g(t) + \Delta f_{P_g}(t) + \nu_{P_g}(t), \tag{2.20}$$

where $\Delta f_{P_g}(t) = (\delta_{f_{P_g}} - 1)\,\tilde{P}_g(t)$ represents the fault term with its loss of effectiveness represented by $\delta_{f_{P_g}}$, where $0 < \delta_{f_{P_g}} \leq 1$, and ν_{P_g} denotes the external measurement noise. $\tilde{P}_g(t)$ represents the added effects of DT oscillations to equation (2.18) as follows:

$$\tilde{P}_g(t) = P_g(t) + \gamma_P \sin(2\pi\sigma_P t),$$

where γ_P and σ_P denote the amplitude and the frequency of the oscillation, respectively.

2. **Fault 2 (Sensor Fault)**: Pitch misalignment may occur due to a faulty sensor that would appear as an incipient fault or due to an incorrect assembly of blades

during installation of the WT, in which this issue would be existed from the beginning of the WT operation. Here, the former scenario (pitch misalignment due to a faulty sensor) is considered. This fault may cause extra fatigue loading on the tower and the blades, resulting in a lower machine life-time [99], [100]. This additive fault is modeled as

$$\beta_m(t) = \beta(t) + \Delta f_\beta + \nu_\beta(t), \tag{2.21}$$

where Δf_β represents the fault bias term ($\Delta f_\beta = 0$ in case of no fault), and ν_β denotes the external measurement noise.

3. **Fault 3 (System Fault)**: This fault occurs when there is a variation in the drive-train damping (equation (2.22)). The drive-train oscillations could be due to wear and tear and give rise to additional fatigue loading on the DT components, particularly at high wind speeds. This fault model is given by

$$\tilde{\omega}_{g,m}(t) = \omega_g(t)\left(1 + \frac{\sin(2\pi\sigma_P t)}{\omega_{g,\max}}[\gamma_\omega + \Delta\gamma_\omega]\right) + \nu_{\omega_g}(t), \tag{2.22}$$

where $\Delta\gamma_\omega = (\delta_{f_{\omega_g}} - 1)\gamma_\omega$ represents the fault term ($\delta_{f_{\omega_g}} = 1$ in case of no fault), ν_{ω_g} denotes the external measurement noise, $\omega_{g,\max}$ denotes the maximal generator speed, and γ_ω denotes the oscillation amplitude coefficient.

Chapter 4 studies the detection and isolation of the above faults, in which an NSA-based FDI scheme is developed for this purpose.

2.4.1.1 Sensor Measurement Faults

The two main types of sensor faults considered are either a *fixed value* or a *gain factor* on the measurements and can be due electrical or mechanical faults in the sensors. The FDI of faults in the pitch angle measurements, $\Delta\beta_{i,m}$, is very important due to their use in the internal controller of the pitch system [13].

Rotor and generator speeds are measured using encoders, which can also become faulty. The encoder may be stuck at a certain value (fixed value fault), and hence will not be updated. In case of a gain factor fault, the encoder counts more marks on the rotating part than the actual value [13]. Faulty rotor and generator speed measurements are denoted by $\Delta\omega_{r,m}$ and $\Delta\omega_{g,m}$, respectively.

The two sensor fault types are modeled as follows:

a. **Fixed value fault**: Equations (2.23) and (2.24) model the fixed value fault in both pitch angle as well as in rotational speed sensor measurements, respectively,

$$\beta_{i,m_j}(t) = \beta_i(t) + \Delta\beta_{i,m_j} + \nu_\beta(t), \tag{2.23}$$

where $\beta_i(t)$ is given by equation (2.6), and $\Delta\beta_{i,m_j}$ represents the fault fixed value term ($\Delta\beta_{i,m_j} = 0$ in case of no fault), and

$$\omega_{p,m_j}(t) = \omega_p(t) + \Delta\omega_{p,m_j} + \nu_{\omega_p}(t), \tag{2.24}$$

where $\omega_p(t)$ ($p = g, r$ in case of generator and rotor, respectively) is given by equation (2.15), and $\Delta\omega_{p,m_j}$ represents the fault fixed value term ($\Delta\omega_{p,m_j} = 0$ in

case of no fault).

b. **Gain factor fault**: Equations (2.25) and (2.26) model the gain factor fault in both pitch angle as well as in rotational speed sensor measurements, respectively,

$$\beta_{i,m_j}(t) = \beta_i(t) + \Delta f_{\beta_{i,m_j}}(t) + \nu_\beta(t), \tag{2.25}$$

where $\Delta f_{\beta_{i,m_j}}(t) = (\Delta\beta_{i,m_j} - 1)\,\beta_i(t)$ represents the fault term, and $\Delta\beta_{i,m_j}$ denotes the gain factor ($\Delta\beta_{i,m_j} = 1$ in case of no fault), and

$$\omega_{p,m_j}(t) = \omega_p(t) + \Delta f_{\omega_{p,m_j}}(t) + \nu_{\omega_p}(t), \tag{2.26}$$

where $\Delta f_{\omega_{p,m_j}}(t) = (\Delta\omega_{p,m_j} - 1)\,\omega_p(t)$ represents the fault term, and $\Delta\omega_{p,m_j}$ denotes the gain factor ($\Delta\omega_{p,m_j} = 1$ in case of no fault).

Chapter 5 studies the detection and isolation of the sensor measurement faults mentioned above, in which a DCA-based FDI methodology is proposed to perform FDI of the sensor faults that are included in the WT benchmark model [13].

2.5 Conclusions

In this chapter, the detailed model of the WT subsystems as well as the wind model are presented. Moreover, the main operational regions of the variable-speed WTs and the corresponding control strategies are explained. Finally, the commonly occurring faults in WTs are presented and the considered faults in this work are modeled.

Chapter 3

Overview of Artificial Immune System

Inspired by the vertebrate Immune System (IS), the field of Artificial Immune System (AIS) has emerged as a promising biologically inspired Computational Intelligence (CI) technique. Important immunological mechanisms that are currently used in the AIS are immune networks, clonal selection theory, Negative Selection (NS) process, and the DAnger Theory (DAT) models, particularly the Dendritic Cell Algorithm (DCA). Immune networks theory is commonly applied in learning and memory, as in the work of [101]. Clonal selection principle has many applications in search and optimization problems [102], as well as in other engineering applications like in [103], and the Negative Selection Algorithm (NSA) as well as the DAT model have gained immense interest

amongst researchers in anomaly and change detection topics [35].

Both the NSA and DCA were developed based on different complex defense mechanisms of the IS in protecting body against any infection. Based on the functionality and type of defense, the IS can be divided into two types that are: *innate immune system* and *adaptive immune system*. Both innate and adaptive immunity work together to get rid of the disease-causing agents, known as pathogens, like parasites, viruses, bacteria, and *etc* [104].

1) **Innate immune system (nonspecific immunity):** The innate IS is the first line of defense against diseases and consists of important immune cells such as DCs, macrophages, and natural killer cells. The innate IS cells respond to pathogens in a generic (non-specific) way by performing an inflammatory response once a Pathogen-Associated Molecular Pattern (PAMP) is recognized. The DCA was developed based on the way DCs operate in the innate IS.

2) **Adaptive immune system (specific immunity):** Adaptive IS consists of lymphocytes (a type of white blood cell), particularly T-lymphocytes (T-cells) and B-lymphocytes (B-cells). These cells of the adaptive IS can distinguish different types of pathogen, and consequently trigger different immune responses against a specific type of pathogen (and hence, the name specific immunity). An important attribute of the adaptive immunity is the self/nonself recognition property that allows the IS to distinguish the self (body's own cells) from the nonself (foreign

invaders). Unlike the innate IS's rapid response (usually in hours), the adaptive IS has a slower immune response (typically in days), particularly in the first encounter of a pathogen. However, the adaptive IS also possess immunologic memory, meaning that in case of the second encounter with the same disease-causing agent, a faster and more effective immune response (referred to as the secondary response) will be induced [105]. The NSA was developed based on the maturation of T-cells in the adaptive IS.

3.1 Negative Selection Algorithm (NSA)

The Artificial Immune System (AIS) is an abstract of the complex vertebrate Immune System (IS), which borrows important mechanisms in the natural IS to computational systems with the main goal of problem solving. An important element of the IS is the T-cells (or T-lymphocytes) that play a key role in the IS.

T-cell is a type of white blood cells that performs important immune functions, such as circulating throughout the body looking for abnormalities. A unique feature of T-cells is their ability to differentiate between normal and abnormal (infected or cancerous) cells in the body.

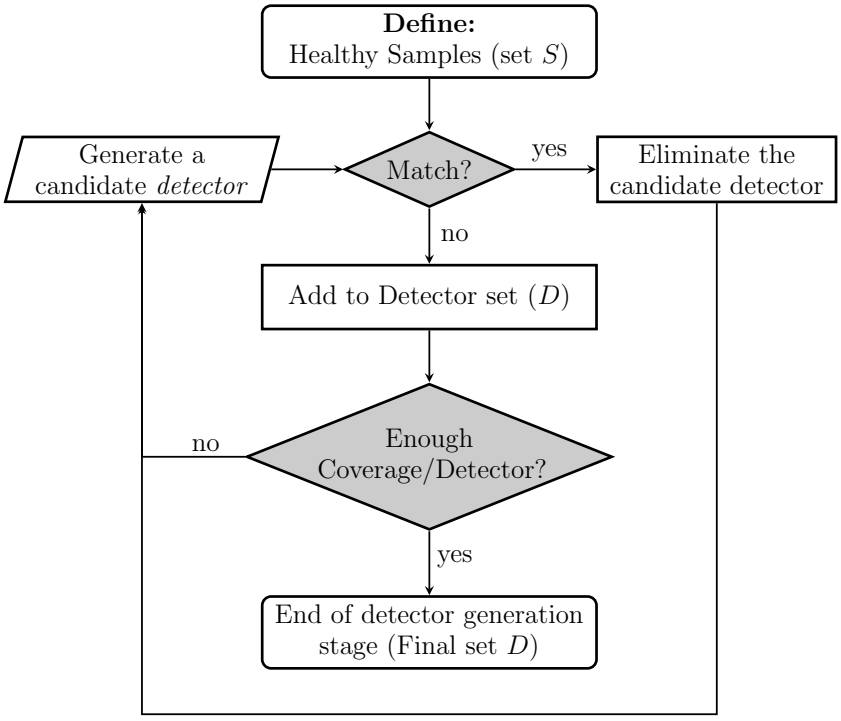
The surface of T-cells is covered by a number of receptors (detector) known as the T-cell Receptor (TCR) that are generated through a pseudo-random genetic rearrangement process [106]. They have the capability of detecting antigens (foreign proteins).

Any protein that can be matched with these TCRs is considered as a malicious object. The T-cells that bind to self-molecules are destroyed, while the ones that do not bind are permitted to leave the thymus [106]. This censoring process of the T-cells in the thymus is known as *negative selection*. This complex mechanism by which the IS differentiates the body's own cells (self) from the foreign cells such as viruses and bacteria (non-self) is known as the Self-Non-Self (SNS) discrimination mechanism.

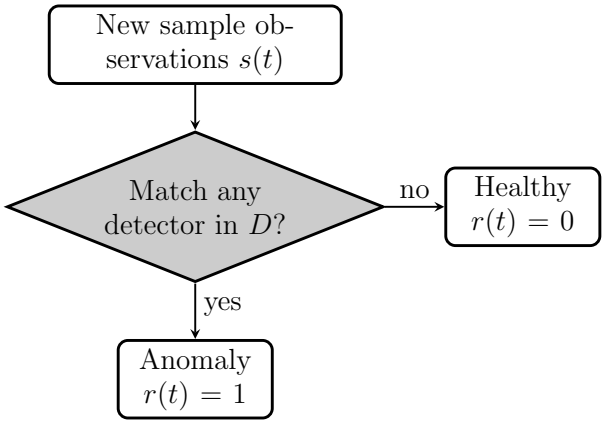
NSA is inspired by SNS discrimination theory and was first introduced in [35]. The body's own molecules are considered to be normal or healthy data (self), while any pathogens such as viruses, fungi, or bacteria are regarded as the faulty data. The main engineering consideration in the NSAs is to cover as much as the non-self region as possible with a minimum number of detectors [107]. The basic mechanism of the NSA can be described as follows [106]:

1. Define the *Self* set (S) that describes the normal behavior of a system that needs to be monitored.
2. Generate a set of detectors (denoted by D), each of which fails to match any element in S.
3. Monitor new observations for any change by matching the detectors in D against the new observations.

The negative selection algorithm involves two stages that are shown in Figure 3.1.



(a) Detector generation (training) phase.



(b) Monitoring phase.

Figure 3.1: The basic NSA methodology and its phases.

First, in the training stage (the first two enumerations), a set of detectors $\boldsymbol{D}$ is generated in which the candidate detectors are trained to be tolerant to the healthy

data samples S (Figure 3.1(a)). In most negative selection algorithms, the candidate detectors are generated randomly and are checked with all the system healthy data in a point-wise manner so that not to match any normal data at the training phase. In the detection (monitoring) phase (Figure 3.1(b)), the new data will be checked with the matured detectors in set D so that any anomaly can be detected [107].

Some of the most important characteristics of NSAs deal with representation schemes, matching rules, and detector generation/censoring processes. Data could be binary (or string), real-valued or hybrid representation of both binary and real-valued. Detectors can be represented by various shapes such as hyper-rectangles [108], hyper-spheres [107], and hyper-ellipsoid [109], and the most common detector representation is the hyper-spheres. One reason for the popularity of hyper-spheres is due to their easier implementation, since they can be represented by a center (a multi-dimensional point) and a radius. Whereas a hyper-rectangular detector is represented by two multi-dimensional data points specifying the lower and upper corners of the hyper-rectangle. It is also difficult to work with hyper-ellipsoidal detectors due to the way they are generally represented, which is based on the orientation of their semi-axes. Moreover, in many engineering applications, data samples (self-set) are represented as hyper-spheres, hence the use of hyper-sphere representation for detectors would be more convenient. In [110], a framework for generating detectors is presented.

The matching rule defines the affinity measure (distance) on the shape space.

Various matching rules have been defined according to the data type. Hamming distance [111], r-contiguous bits [35], and r-chunk matching [112] represent as some of the binary matching rules that are used with binary data [113]. The most commonly used matching rule associated with the binary data is the r-contiguous matching [63].

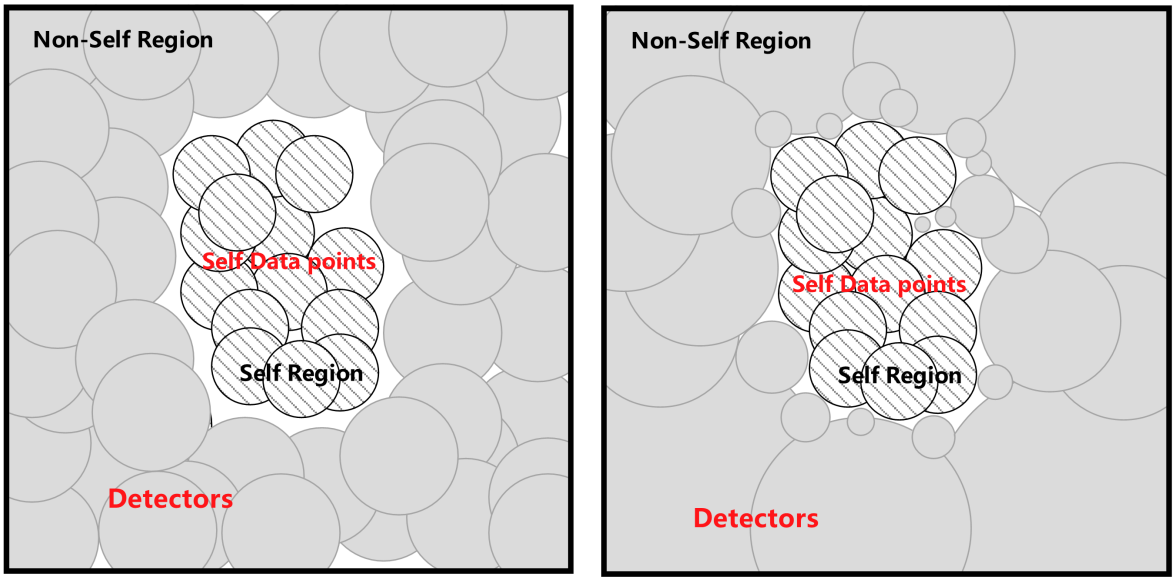
In case of real-valued data, Manhattan distance, infinity norm distance, and most commonly the Euclidean metric are utilized to represent the data and detectors [107]. The detector generation/elimination procedure represents as another important characteristic of the NSA and can be implemented by using one of the following approaches [113]:

- Random generation of candidate detectors followed by the censoring mechanism,
- Genetic algorithm,
- Greedy algorithm or other deterministic algorithms.

In this work, due to the real-valued nature of data samples that are collected from the WT measured variables, and the associated real shape-space, a real-valued NSA is employed. Data points are represented as hyper-spheres with fixed self-radius ($\boldsymbol{R}_{\text{self}}$) that determine their thresholds. However, there is no universal mean to decide on the particular value of the self-radius, and this is commonly determined empirically [114].

The main objective of the NSAs is to cover the entire non-self region with detectors that are represented as spheres with either fixed-radius or variable-radius. As an illustration, and without loss of generality, a 2-dimensional representation of shape space with

constant-sized (fixed-radius) and variable-sized (variable-radius) detectors are shown in Figures 3.2(a) and 3.2(b), respectively. It follows readily that variable-sized detectors have better performance in covering the non-self region, particularly on the boundary between the self and the non-self regions. By utilizing variable-radius detectors, fewer detectors are needed to cover the non-self region, in comparison with the fixed-radius detectors.



(a) Constant-sized detectors.

(b) Variable-sized detectors.

Figure 3.2: The main NSA concept having different detector radii (motivated from [107]).

In this thesis, a real-valued NSA known as the *V-detector* [107, 115] (originally developed in [116]) is selected and implemented. In this algorithm, the size of the detectors covering the non-self region is variable (and hence the label variable-detector or V-detector). *The distance measure used is the Euclidean distance and data samples are considered to be spheres.* The objective, as stated above, is to cover as much of

the non-self region as possible. The flowchart describing the details on the V-detector algorithm is depicted in Figure 3.3.

In the V-detector algorithm, first a candidate detector x (represented by the pair (d_c, R_d)) centered at point d_c with a radius R_d is randomly generated. Then, the Euclidean distance, Γ_d , between x and every detector (d_j, R_j) in the detector set D is calculated. If x is detected by another detector in D ($\Gamma_d < R_d$), then the detector x is eliminated. If not, the Euclidean distance, Γ_s , between x and every sample s_i in the self set S is evaluated. It should be noted that all samples have a fix self-radius denoted by R_{self} . If $\Gamma_s > R_d + R_{\text{self}}$, meaning that there is no interference between the sample s_i and the candidate detector x, then x is added to the detector set D. However, if $\Gamma_s < R_d + R_{\text{self}}$, meaning that the sample is interfered with x, then the detector radius R_d is shortened to $R_d = \Gamma_s - R_{\text{self}}$. If the shortened detector radius is negative, the candidate detector x is eliminated and the whole process starts all over again. If not, the candidate detector x with the shortened radius is added to set D. Next step is to check whether the number of detectors in D is greater than preset maximum number of detectors T_{max} . If so, the detector set D is finalized. However, if the number of detectors in D is less than T_{max}, then another criteria is checked and that is whether the minimum coverage is achieved or not. If the minimum coverage of the non-self space is achieved, then the detector set D is finalized and will be ready for the monitoring phase. If not, then a new candidate detector is generated and the whole process repeats.

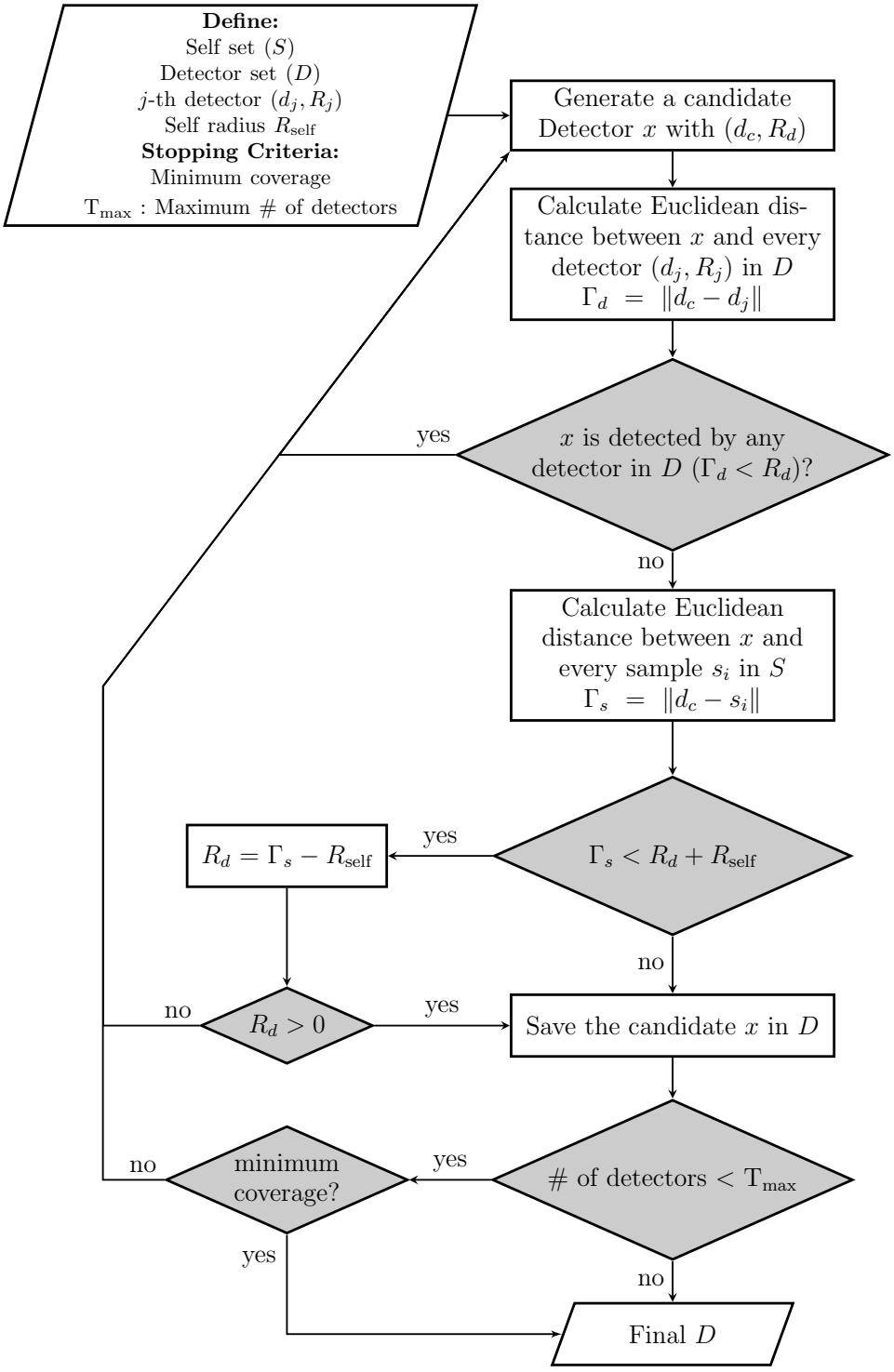


Figure 3.3: The flowchart of V-detector generation algorithm (the candidate detector x (d_c, R_c) is centered at d_c with the radius R_d).

3.2 Dendritic Cell Algorithm (DCA)

For decades, immunologist thought that the IS differentiates between self molecules and non-self molecules coming from outside of the body. This was based on the idea of *SNS discrimination theory* stating that the vertebrate IS has the capability of distinguishing the body's own cell (self) from any foreign cells (non-self). Therefore, based on the SNS models, *foreignness* of a particular cell is the reason that leads to stimulating the immune response. However, IS as described by SNS theory offers no explanation for the phenomena such as transplantation and autoimmunity (immune system response against its own body cells) [117].

An immunologist Polly Matzinger proposed DAT [117] and offered an alternative explanation for how immune systems work. Based on the DAT, any damage to the body triggers the IS by sending danger signals, and in the case of absence of a danger in a tissue, the innate IS can suppress the immune response [65].

3.2.1 Overview on Dendritic Cell Biology

Dendritic Cells (DCs) are considered as professional phagocytes that play an important role in the IS. Phagocytes are white blood cells that protect the body by engulfing foreign particles such as bacteria, dirt, *etc*, and then digesting them as illustrated in Figure 3.4.

In DAT, the task of DCs is to collect, process, as well as present the antigen to the T-cells. Figure 3.4 shows the overall process of the maturation of a DC and

how a pathogen triggers the innate IS. Under healthy conditions, if body cells are no longer needed, they may commit suicide during a controlled process called *apoptosis* or *programmed cell death* to regulate the growth and development of the cells. Due to apoptosis, immuno-suppressive molecules (referred to as *Safe Signals*) are released indicating that all is normal in the tissue, and consequently promoting immune tolerance [118]. The phenomenon of apoptosis is shown in the top part of Figure 3.4. In contrast, when cells are under stress due to an injury or a damage in the tissue, they die during a process called *necrosis*, which is usually caused by external factors such as trauma, infection, *etc.* If a cell undergoes through necrosis, it will burst and release *Danger Signals* [118]. This phenomenon is shown in the bottom part of Figure 3.4.

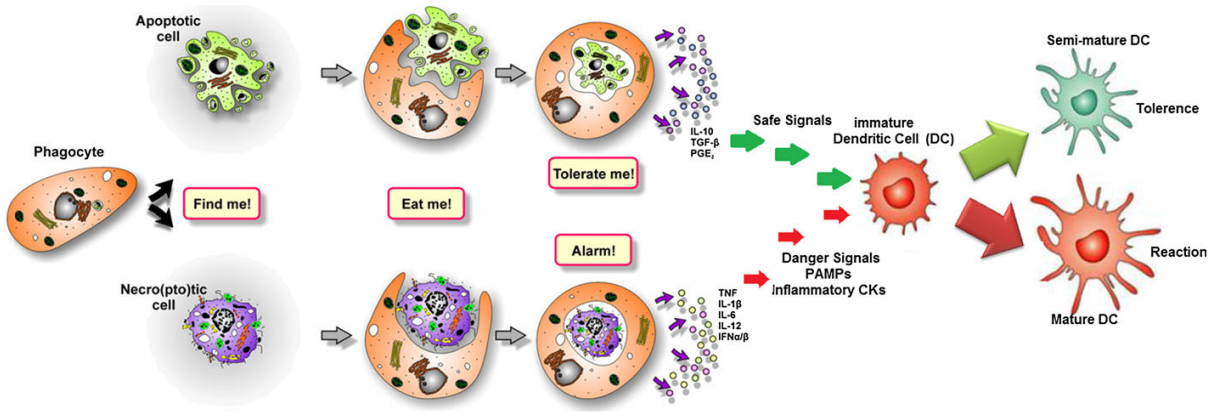


Figure 3.4: Apoptosis and necrosis and DCs (reprinted from [118]).

Moreover, DCs have the capability to sense the danger signals (released as a result of necrosis), safe signals (released as a result of apoptosis), and PAMPs are molecules associated with pathogens such as virus, fungi, bacteria, *etc*). DCs are also sensitive

to inflammatory signals originating from the tissues in case of inflammation due to an infection or tissue injury. However, inflammatory signals only amplify the effects of other three signals and will not have an effect on DCs behavior if they are used alone. The four main immunological signals are briefly summarized below [79]:

- **PAMP Signal (PS)**: Released in case of presence of any pathogen (such as bacteria).
- **Danger Signal (DS)**: Released in case of an unplanned cell death (necrosis).
- **Safe Signal (SS)**: Released in case of a planned cell death (apoptosis).
- **Inflammatory Cytokine (Inflammatory Signal)s (ICs)**: ICs (also called *inflammatory signals*) are released when there is an inflammation in a tissue.

The behavior of DCs and their maturation level depends on the concentration of the immunological signals mentioned above. DCs always exist in one of the three maturation states at any given time [104]:

- **Immature DC (iDC)**: iDCs are the initial maturation state of DCs, in which they collect the transmitted immunological signals, and based on their concentration, iDCs change their maturation level to either a partially maturated state (and hence called semi-mature DC) or to a fully maturated state (and hence called mature DC).
- **Semi-mature DC (smDC)**: iDCs become partially mature (or semi-mature) if they are exposed to higher amount of SSs than PSs and DSs.

- **Mature DC (mDC)**: iDCs become mature DCs (mDCs) if they are exposed to a higher amount of PSs and DSs compared to SSs.

Figure 3.5 illustrates the DC's mechanism in the IS. The sampled input signal and antigen are fed into the selected iDC from the DC pool. Interleukin-10 (IL-10) and interleukin-12 (IL-12) are also referred to as the semi-mature and the mature output signals, respectively. Upon exposure to high concentrations of the IL-10 and IL-12 , an iDC matures into the smDC and the mDC, respectively. The other DC output is the Co-Stimulatory Molecule (CSM), which is responsible for the DC migration when the DC reaches its migration threshold $\mathbb{M}$. A graphical illustration of the signal processing within a DC is shown in Figure 3.6, in which the DC outputs are updated based on the concentration of the input signals. It should be noted that each DC in the DC pool has an assigned lifespan $\mathbb{L}$, such that when the $\mathbb{L} = 0$, the DC will be reset and joins the DC pool. If the amount of the IL-12 (the mature output signal) is greater than the amount that IL-10 is produced (so that IL-12 $\geq$ IL-10), then the migrated DC (referred to as the *stimulatory DC*) provides the reactive Cytokines by stimulating the killer T-cells (Cytotoxic T-cells) in the adaptive IS. Otherwise, the migrated DC will regulate the T-cells (referred to as the suppressor/regulatory T-cells) in adaptive IS by suppressing the T-killer cell. If at the end, there are killer T-cells, an immune response would be initiated, otherwise, an immune tolerance will be invoked.

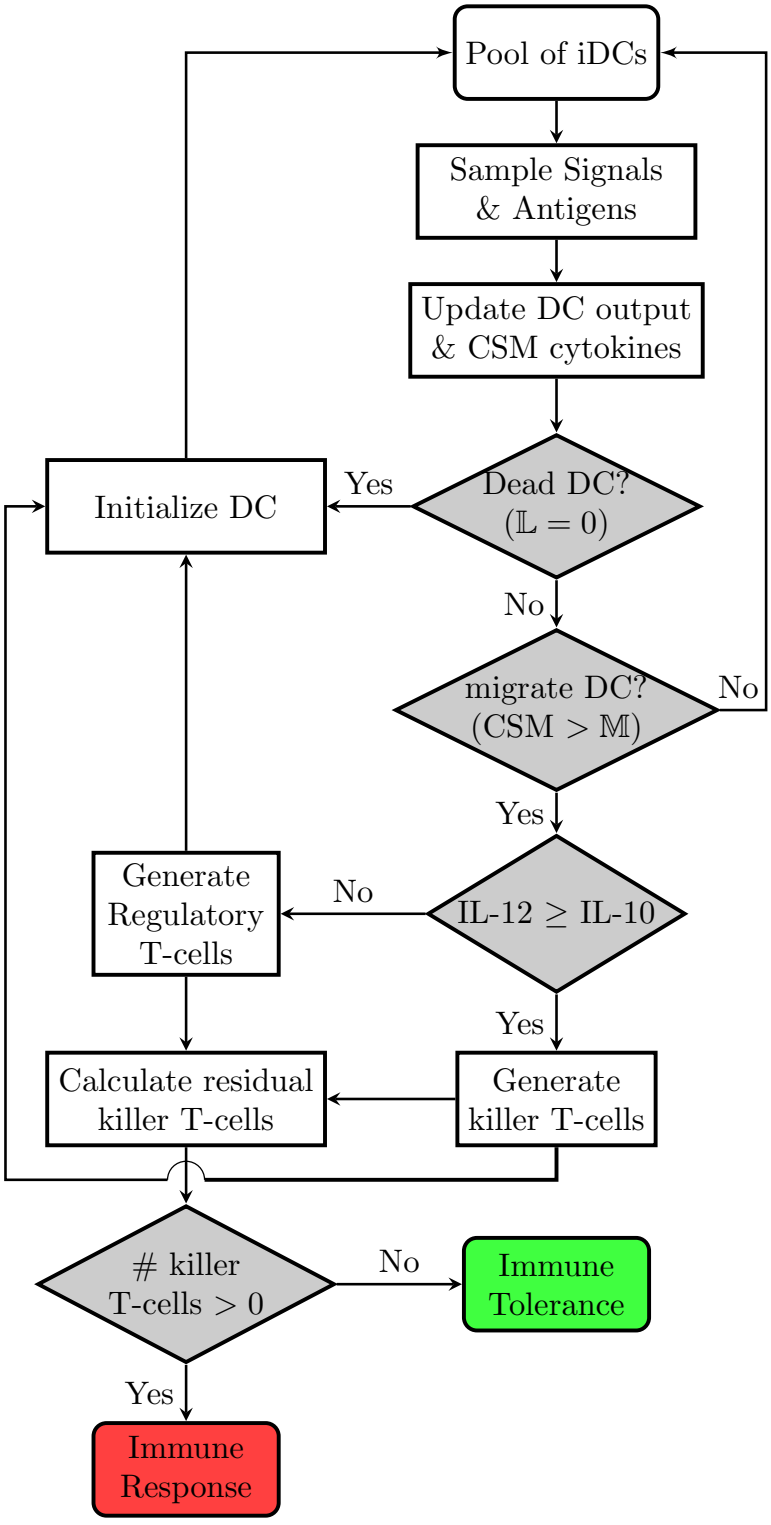


Figure 3.5: Flowchart of how a DC works in the immune system.

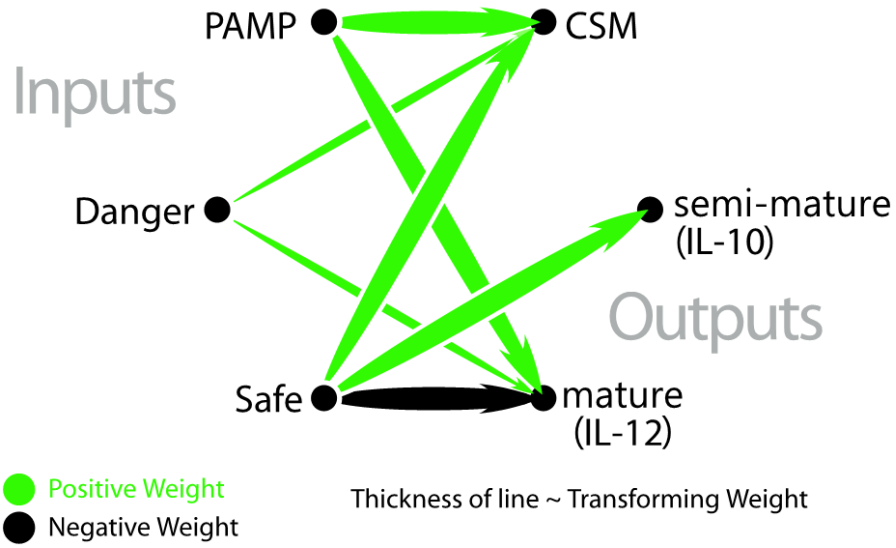


Figure 3.6: The signal processing of a DC model (adapted from [65]). The IC signal is not shown in the model.

3.2.2 The Dendritic Cell Algorithm (DCA)

The Dendritic Cell Algorithm (DCA) is developed based on an abstract model of the DC biology (explained in Section 3.2.1). The DCA is first introduced by Greensmith [65] as part of the Danger Project [66] with the goal of applying a paradigm of state-of-the-art immunology to the problem of the computer network Intrusion Detection System (IDS).

The DCA is based on a pool of DCs, making it a *population-based* algorithm with each artificial cell acting as an agent within the system (unlike most negative selection-based algorithms that are *instance-based*). In [119], a frequency analysis of a single DC was performed in order to analyze the flow of information through the DCA. According to [119], a single DC can be modeled as a low-pass filter.

The most important step of any DCA version is the choice of the input signals to the algorithm. The DCA usually does *not* require any training phase. Hence, if expert knowledge is not available or the selection of input signals for the DCA is not trivial, feature extraction or statistical inference methodologies can be utilized to obtain the most suitable features for DCA input signals.

There are two types of DCA input data: *signals* and *antigens*. Signals (explained in Section 3.2.1) are vectors of real-valued numbers indicating the status of the observed system. There are three time-varying signals, namely PS, DS, and SS, in addition to the IC signal. The semantic of these biologically inspired signals is as follows:

- **PAMP Signal (PS)**: A signal that indicates the occurrence of an abnormal behavior. An increase in PS is linked to a *high confidence of abnormality*.
- **Danger Signal (DS)**: A signal that increases in case of an anomalous situation with a less confidence level of abnormality as compared to the PS.
- **Safe Signal (SS)**: A signal that indicates a normal behavior, and its value increases in conjunction with the predictable normal behavior of the system.
- **Inflammatory Signal (IC)**: A signal that amplifies the effects of other immunological signals and has no effect if it is used in isolation.

More details on the background of different signals are explained in Section 3.2.1.

Another type of input data is antigen without which the DCA will not be able to function. Antigens are categorical values corresponding to entities of a monitored

system that need to be classified [120].

Each DC within the DC pool has a limited lifespan (related to CSM value) upon its creation. Following that, the DC migrates to a processing center known as the lymph node, where the DC presents the antigens coupled with the output signals that has already been collected during its lifespan [65]. Based on the context of the signals, the DC differentiates into either: smDC which suppresses the immune response, or mDC that stimulates the immune response.

The signal processing in each DC is expressed as follows

$$\mathrm{DC}_{\mathrm{out}_j} = \frac{1 + \mathrm{IC}}{2} \times \sum_{i=0}^{2} (W_{ij} S_i), \quad j = 0, 1, 2 \tag{3.1}$$

where

- W_{ij} denotes the signal weights connecting the i^{th} input to the j^{th} output,
- i is the input signal category (PS when $i = 0$, DS when $i = 1$, and SS when $i = 2$)
- j is the output signal value where
  - $j = 0$ is a Co-Stimulatory Molecule (CSM),
  - $j = 1$ is a smDC output signal,
  - $j = 2$ is a mDC output signal.

In most DCA versions, the effects of the IC signal are ignored (*i.e.*, IC = 1).

The overall procedure of the DCA is given in the Pseudocode 1. First, a signal database is created through fusing the input signals with antigens, and this task is

achieved through the functions *get antigen()*, *store antigen()*, and *get signal()*. Then, the function *calculate output signals()* evaluates the output signals for each DC using the input signals (PS, DS, and SS) and a set of pre-defined weights based on equation (3.1). The objective is to classify the antigens based on output of a population of DCs. As a given DC receives more input signals, its level of co-stimulatory signal (known as Co-Stimulatory Molecule (CSM) that is related to the DC's lifespan) is increased. Once the CSM reaches the *Migration Threshold*, the DC stops collecting (sampling) input signals as well as antigens, and migrates to the lymph nodes where the collected antigens are presented. Based on the antigens presentations, each DC forms a *cell context* that is later used to calculate the anomaly metric for each antigen type (or equivalently, to classify the antigens). After presenting the antigen, the DC is removed from the population and is replaced by a new cell. More detail on the mechanism of the DCA is available in [79].

The original DCA that was introduced in [65] has many parameters and stochastic elements that made it difficult to be implemented. The same authors proposed a deterministic version of the DCA known as deterministic DCA (dDCA) in [67] and tested the algorithm to validate its performance. In [74], a modified DCA (*mDCA*) was introduced with the aim of an online error detection in robotic systems. The proposed DCA-based framework is based on modifications of and extensions to the mDCA.

Pseudocode 1: DCA algorithm [121].

Input: Antigen and Signals
Output: Anomaly Metric
Initialization
while *CSM < Migration Threshold* **do**
    get antigen();
    store antigen();
    get signal();
    calculate output signals();
    update CSM();
end
if *smDC > mDC* **then**
    cell context = 0;
else
    cell context = 1;
end
kill the cell;
replace the cell in the population;
for *each Antigen type* **do**
    Calculate the anomaly metric
end

3.3 Conclusions

In this chapter, an overview on the way human Immune System (IS) operates and its different defense techniques are provided. In addition, two important mechanisms inspired from the Artificial Immune System (AIS), namely the Negative Selection Algorithm (NSA) and Dendritic Cell Algorithm (DCA) are presented, and the background as well as the mechanism of both algorithms are explained in details.

Chapter 4

NSA-based FDI Scheme

In this chapter, an FDI methodology based on the Negative Selection Algorithm (NSA) is proposed. The proposed methodology is then applied to the wind turbine benchmark model in order to illustrate the capabilities of the proposed scheme. A comparison between the NSA-based FDI scheme and a well-known data-driven technique, namely the Support Vector Machine (SVM) is made. Moreover, a non-parametric statistical comparison test is conducted to compare the performance of the developed NSA-based FDI methodology with the SVM.

4.1 Proposed NSA-based FDI Scheme

The block diagram depicting the main steps of the proposed NSA-based FDI scheme is shown in Figure 4.1. The aim of the *Preprocessing Data* block is to process the input

data through feature selection, sampling, and normalization to be used for the NSAs. Important features of the system should be carefully chosen to ensure crucial system characteristics are observed, so that when a fault occurs in the system, the selected features do capture the system abnormal behavior from which the FDI task can be accomplished. All the candidate detectors (that are spheres in this chapter) should be checked with every data sample in the self set S to ensure that the candidate detector does not cover any data sample. Therefore, data sampling is implemented to obtain a subset of the original data for reducing the runtime complexity of the NSA training stage. Data normalization is a crucial step in data preprocessing for implementation of real-valued NSAs. The data is generally normalized within the range of $[0, 1]$ or $[-1, 1]$. In this work, the WT measurement outputs are normalized within the range of $[-1, 1]$ before feeding them to the NSAs.

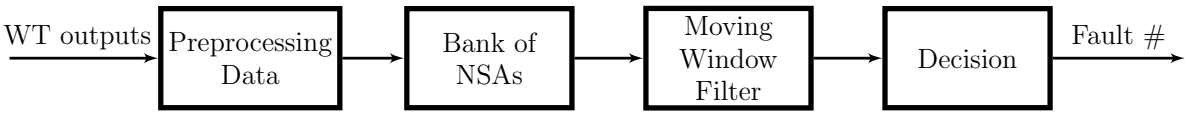


Figure 4.1: The main steps for implementation of the proposed NSA-based FDI scheme.

A summary of the most commonly developed fault diagnosis techniques in the literature versus the proposed NSA-based FDI scheme is given in Table 4.1.

Table 4.1: A comparison of the proposed NSA-based FDI methodology against other FDI techniques that are applied to the benchmark WT model [13].

Methodology	**Detection & Isolation**	**Linear / nonlinear Approach**	**Simultaneous faults**	**Isolation of system faults (due to drive-train vibrations)**
Estimation-based approach ([16, 19, 34])	✓	Linear	×	× (except [34])
Set-Membership method ([20, 21])	✓	Nonlinear	✓	× (except [20])
Robust FDI filter [122]	✓	Linear	×	×
Support Vector Machine ([28, 30])	✓	Nonlinear	×	×
Fuzzy method ([25, 27])	✓	Nonlinear	×	✓
Proposed NSA-based FDI method	✓	Nonlinear	✓	✓

In the proposed FDI scheme, a number of NSAs are configured in multiple layers to implement the FDI tasks as shown in Figure 4.2.

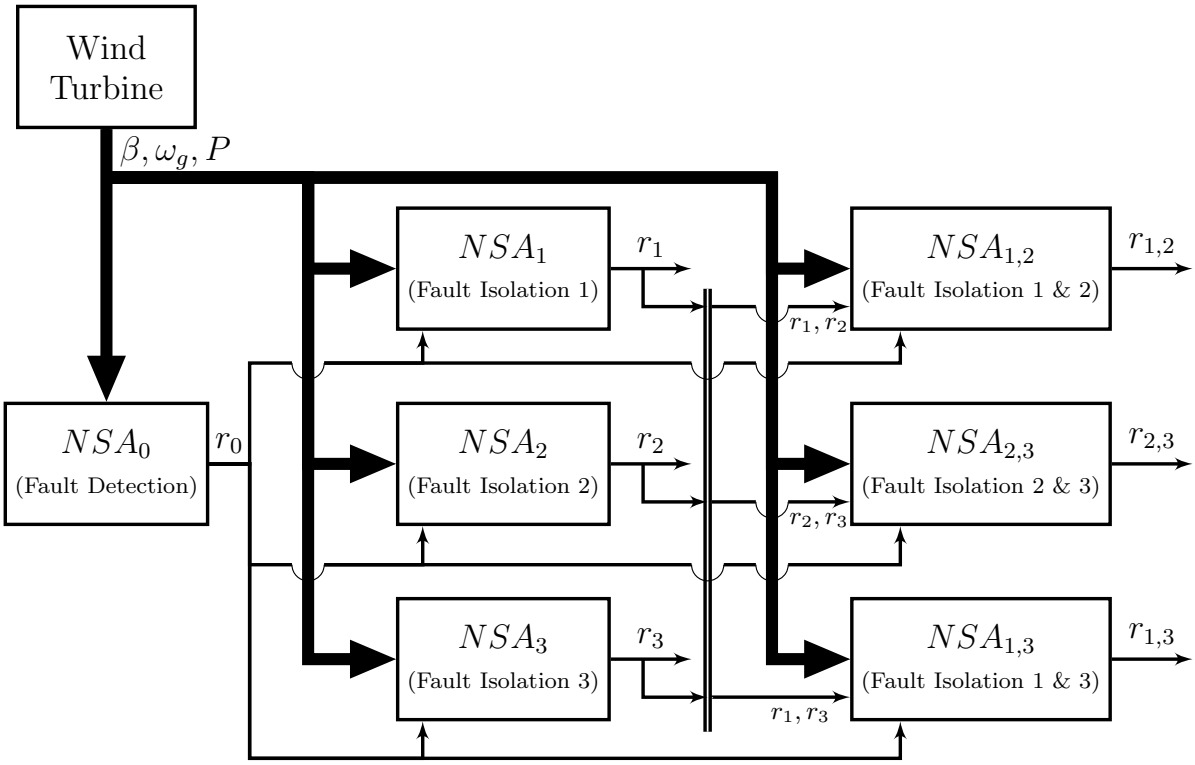


Figure 4.2: The bank of NSAs for the proposed NSA-based FDI scheme.

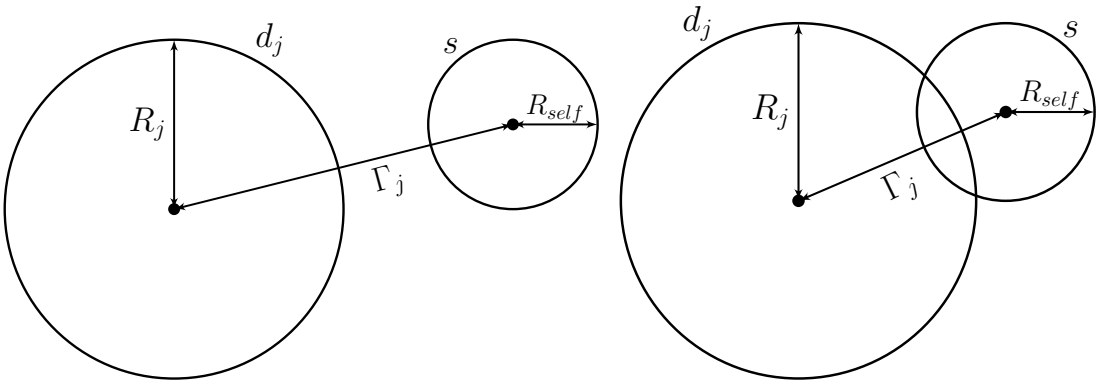
Each NSA consists of center and radius of the spheres (detectors) that cover the non-self region of the shape space. For example, NSA_0 contains details on all detectors (the radius and center of all the spheres) covering the shape space of the non-self region associated with the healthy condition. Each NSA is generated and trained based on the flowchart that is shown in Figure 3.3. For each data sample $s(t)$, the residual signal $r(t)$ that is generated by an NSA is given by

$$\Gamma_j(t) = \left\| s(t) - d_j \right\|, \tag{4.1a}$$

$$r(t) = f(s(t)) = \begin{cases} 1 & \exists\ d_j \in D \ \text{ s.t. } \ \Gamma_j(t) - R_j < R_{\text{self}}, \\ 0 & \text{otherwise}, \end{cases} \tag{4.1b}$$

where d_j denotes the j-th detector in the detector set D, R_j denotes the radius of the j-th detector, and $\Gamma_j(t)$ denotes the Euclidean distance between the j-th detector and the data sample $s(t)$. Figure 4.3 depicts the Euclidean distance between the detector and the sample under both healthy as well as faulty scenarios. For simplicity, and without loss of generality, the Euclidean norm is illustrated in the two-dimensional space.

For performing the fault detection task, the first NSA, namely the NSA_0 is trained to detect any abnormality in the WT system and to remain insensitive to only normal and healthy samples that are generated by the WT model. However, given that the other goal is to isolate a fault, therefore for performing this task a parallel bank of NSAs is constructed where each NSA is trained to be insensitive to a specific and a



(a) Healthy condition ($r(t) = 0$). (b) Faulty condition ($r(t) = 1$).

Figure 4.3: The sample-detector distance (monitoring phase).

particular fault only. The first layer in the bank of NSAs (namely, NSA_1, NSA_2, and NSA_3) isolates the faults that *do not occur simultaneously*. This layer is activated only when a given fault is detected by NSA_0. Given this scenario, the fault residual in this case is denoted by $r_0(t) = 1$. The second layer of NSAs (namely, $\text{NSA}_{1,2}$, $\text{NSA}_{2,3}$, and $\text{NSA}_{1,3}$) is responsible for isolating *concurrent faults*, and is trained to be insensitive to multiple faults. For example, $\text{NSA}_{1,2}$ is trained to isolate the *simultaneous* occurrence of Faults 1 and 2. It should be noted that for training the bank of NSAs, various fault severities are considered in generating the detector set for each NSA.

For eliminating outliers as well as for improving the performance of the binary residuals that are generated by the NSAs, a *moving window filter* is applied to the fault residual signals. This filter will move and slide (with a particular given window size) over the fault residuals that are generated by the NSAs. Depending on the number of faulty samples within each window, a decision is rendered to as if all samples in the

window should be categorized as faulty or not. Depending on the type of fault, different window filter parameters will be utilized. For the Faults 1 and 2, a moving window with a length of 0.2 seconds is used, and if more than 75% of samples in the window are at the high state (that is, when $r_1(t) = 1$ or $r_2(t) = 1$), the entire window will be set to 1. However, for the Fault 3, the same window length is chosen but with a lower threshold. If at least 50% of the residuals in the window is faulty, the entire window will be set to 1. The reason for this change has to do with the nature of the fault type, which is shown as a change in the amplitude oscillation of ω_g. The parameters of the window filter (namely, the window length and the threshold) are tuned empirically according to the fault detection and false alarms performance results.

Depending on the characteristics of the residuals generated by NSAs, a decision is made as to whether a fault has occurred or not; and in case of a fault declaration, its type is also determined. The fault detection and isolation logic can therefore be stated as follows.

Fault Detection and Isolation Logic

Assuming that a fault is detected by NSA_0 (that is, $r_0(t) = 1$), the decision for isolating the faults based on the first layer of NSAs is made as follows: If the i-th residual signal is zero, *i.e.*, $r_i(t) = 0$, while the other residuals are active, that is $r_j(t) = 1$, $j \neq i$, it is then concluded that the fault i has occurred. The same logic and reasoning can also be

extended for isolation of simultaneous faults by utilizing the second layer of NSAs. In this case if $r_j(t) = 1$ and $r_k(t) = 1$, and only $r_{j,k}(t) = 0$, then the faults j and k are simultaneously detected and isolated.

4.2 Simulation Results

In this section, the results for implementing the proposed NSA-based FDI scheme corresponding to both non-simultaneous and simultaneous fault scenarios are presented. The results are then compared with another data-driven approach known as the SVM [123]. Moreover, a non-parametric statistical test, known as the *sign test*, is implemented in a pairwise manner (pairwise comparison) for comparing the performance of the proposed NSA-based methodology with the SVM under various fault cases that range from low fault severities to high fault severities. 25 Monte Carlo simulation runs are conducted for each fault scenario corresponding to randomly selected fault severities. However, for illustration purposes, only results for certain fault severities are provided for each fault scenario.

Before demonstrating the capabilities of the proposed NSA-based methodology, it is necessary to provide details on the properties of the NSAs that are used in the FDIs scheme. The features specifying the shape space are chosen to be outputs of the WT system (namely, ω_g, β, and P_g). This represents a 3-dimensional space. Correspondingly,

the data points are represented as spheres with self-radius (threshold) of 0.1 that is selected experimentally. Detectors are also chosen to be sphere shaped and are generated via the V-detector algorithm in which the candidate detector centers are generated randomly. The data samples from the outputs of the WT system along with the generated detectors in the shape space are shown in Figure 4.4, where the red spheres represent the normal WT system data that construct the self region and the blue spheres are the NSA detectors covering whatever is outside the self region (that is the non-self region).

Before implementing the real-valued NSA, as stated earlier it is necessary to normalize the data. For generating the detectors for each NSA, 80% of the data set (the WT outputs) is used for training and the other 20% is allocated to the testing phase. In this work, the maximum number of detectors chosen is set sufficiently large to not terminate the detector generation process too early. Also the termination condition provided in Figure 3.1(a) is set to 99% coverage of the non-self space for the maximum of 10000 detectors. Therefore, the training phase is completed only when the desired coverage of the non-self region is attained. The justification behind selecting the particular coverage percentage above is that choosing a higher coverage results in a much higher processing time as well as a dramatic increase in the number of the detectors that have to be utilized albeit for a marginally and not too significant higher coverage yield.

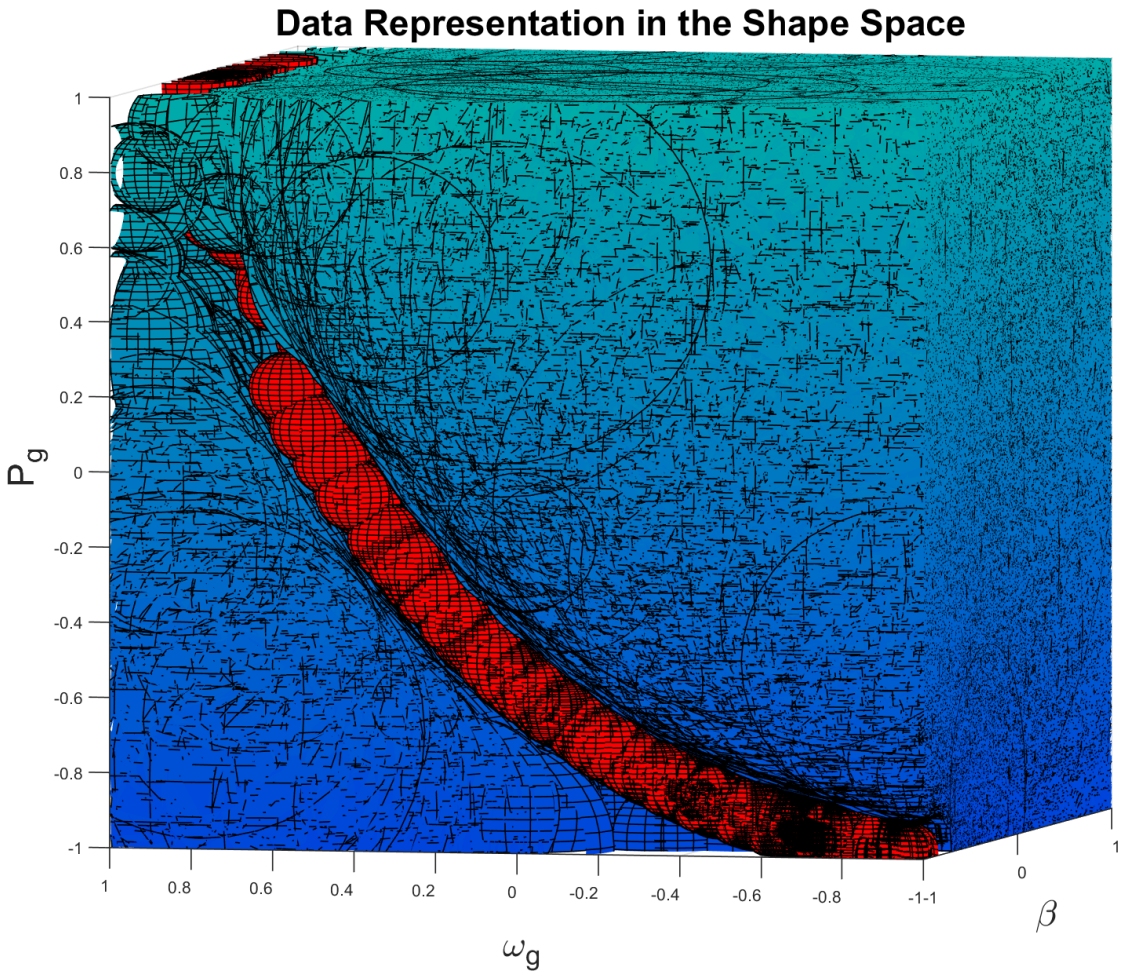


Figure 4.4: The WT outputs and the detectors in the shape space (NSA_0). Red spheres: WT healthy data and blue spheres: detectors generated to cover the non-self region.

4.2.1 Fault Scenario 1: Non-Simultaneous Faults

For illustrating and evaluating the performance of the fault detection scheme (NSA_0) as well as the fault isolation of non-simultaneous faults (through the use of NSA_1, NSA_2, and NSA_3), three scenarios are considered as described below.

The faults are injected at different time instances as follows:

- **Fault 1** has a loss of effectiveness factor of $\delta_{f_{P_g}} = 0.85$ (refer to equation (2.20)) during the time interval 2200s to 2300s.
- **Fault 2** has an offset of $\Delta f_\beta = 1.0°$ (refer to equation (2.21)) during the time interval 1200s to 1300s.
- **Fault 3** has a damping factor of $\Delta\gamma_\omega = 15$ (refer to equation 2.22)) during the time interval 3000s to 3100s.

The measured WT outputs corresponding to the healthy situation as well as when the WT is subjected to presence of all faults (non-simultaneous) are shown in Figure 4.5.

The objective of the first layer of NSAs, as shown in Figure 4.2, is to isolate non-simultaneous faults and this task is clearly accomplished as shown in Figure 4.6. For example, the $\text{NSA}_\mathbf{1}$ considers Fault 1 to be its normal data since $\text{NSA}_\mathbf{1}$ is trained to be insensitive to this particular fault while sensitive to all the other faults. By comparing the fault signatures that are corresponding to $\text{NSA}_\mathbf{0}$ and $\text{NSA}_\mathbf{1}$ in Figure 4.6(a), it follows that Fault 1 is isolated since $\text{NSA}_\mathbf{1}$ is exposed to faulty samples during the period 2200s-2300s as healthy, while the $\text{NSA}_\mathbf{0}$ has already reported that the WT is faulty. Figure 4.6(a) also shows that the $\text{NSA}_\mathbf{3}$ did not detect any faulty samples (missed detections) during the period 2255s-2260s. A similar conclusion can be drawn for the isolation of Fault 2 and Fault 3 based on the results that are shown in Figures 4.6(b) and 4.6(c), respectively. By comparing the fault residuals corresponding to $\text{NSA}_\mathbf{0}$ and $\text{NSA}_\mathbf{2}$ in Figure 4.6(b), it follows that the Fault 2 is isolated. In case of isolating the Fault 3

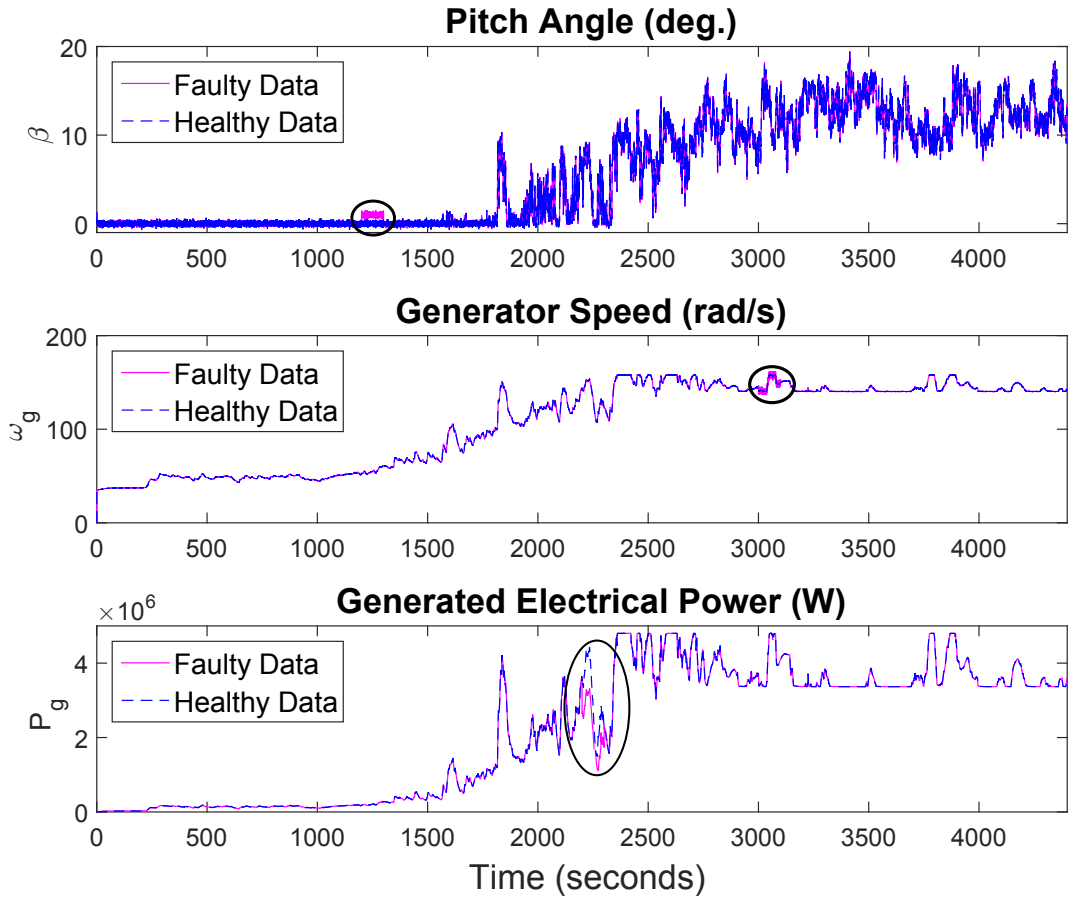
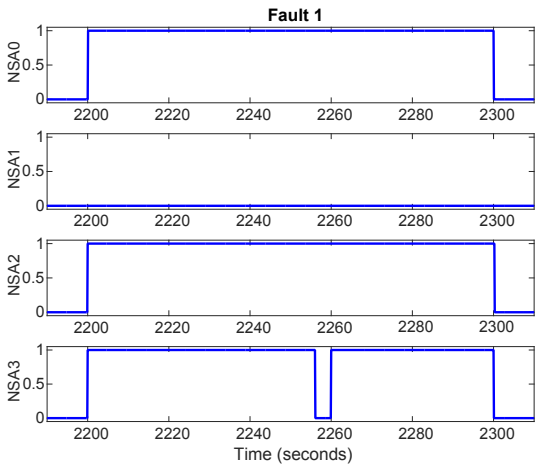


Figure 4.5: The WT measured variables under the healthy and faulty scenarios.

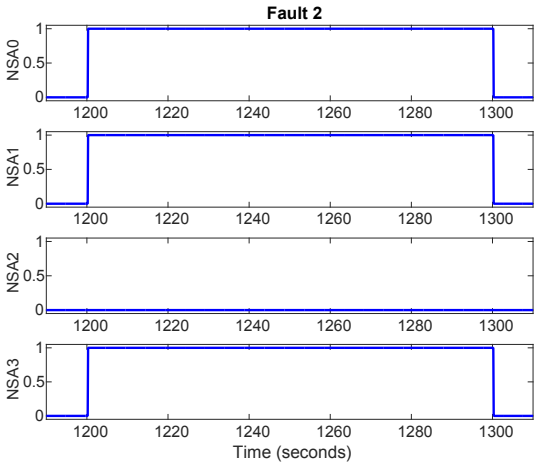
(Figure 4.6(c)), NSA_0, NSA_1, and NSA_2 have missed to detect few faulty data samples, and NSA_3 has few false alarms.

4.2.1.1 Comparison with the Support Vector Machine (SVM)

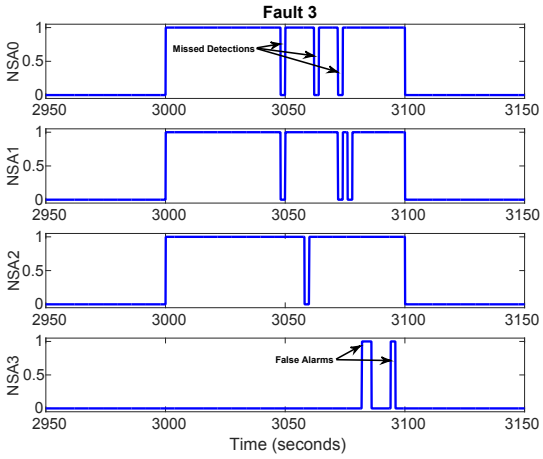
The proposed NSA-based scheme is now compared with another data-driven method, namely the Support Vector Machine (SVM) [123], that is based on structural risk minimization, and hence has improved generalization properties. More detail is available in



(a) The FDI results for the Fault 1 by using the NSA.



(b) The FDI results for the Fault 2 by using the NSA.



(c) The FDI results for the Fault 3 by using the NSA.

Figure 4.6: The FDI results for non-simultaneous faults by using the NSA.

Appendix A. SVMs are widely used in classification problems, however it has also been applied to FDI of wind turbines in [28] and [30], producing good results.

In order to compare the performance of the proposed NSA-based FDI scheme with the SVM, first the FDI problem is reformulated as a classification problem [124]. For performing the FDI task, three SVMs are utilized where they are binary coded. Since all WT measurements are used in developing the bank of NSAs, the same measurements (ω_g, β, and P_g) are also utilized in developing the SVM models. A polynomial kernel function (4.2) is used in these models. Specifically, the kernel function is defined as

$$K_d(\mathbf{u}, \mathbf{v}) = (r + a\mathbf{u}^{\mathrm{T}}\mathbf{v})^d, \qquad a > 0, r > 0 \tag{4.2}$$

where d denotes the degree and (a, r) are the parameters of the polynomial kernel function.

The denoted SVM_1, SVM_2, and SVM_3 are trained to detect the Faults 1, 2, and 3, respectively. A polynomial kernel function of degree 3 is utilized to build the SVM_1 and SVM_2 models, whereas a polynomial function of degree 5 is used for training the SVM_3. The reason for choosing a higher degree polynomial kernel for SVM_3 is due to the more difficult task of performing the FDI of the Fault 3 when compared with Faults 1 and 2. Consequently, a higher degree polynomial kernel is utilized in order to have a more flexible decision boundary, and hence, a better FDI of the Fault 3.

For each SVM, 80% of the data set is allocated for training (out of which 20% is dedicated for cross-validation) and the other 20% is allocated for the testing sets. In

case of presence of a fault, the SVM produces an output of +1, whereas if the WT operates normally (that is, healthy), the SVM produces an output of -1. The SVM outputs corresponding to the three fault cases, indicating the actual WT state as being either healthy (-1) or faulty (1) are shown in Figure 4.7.

Comparative results between the NSA and SVM methodologies in terms of the classification accuracy is evaluated under various fault severities where the mean results are provided in Table 4.3. Specifically, 25 randomly selected fault cases ranging from low fault severity to high fault severity are considered and tested. In case of Fault 1, the fault severity $\delta_{f_{Pg}}$ is randomly varied within $0.25 \leq \delta_{f_{Pg}} \leq 0.95$. In case of Fault 2, the fault bias term Δf_{β} is randomly varied within $0.5^{\circ} \leq \Delta f_{\beta} \leq 10^{\circ}$, and in case of Fault 3, $\Delta \gamma_{\omega}$ is randomly varied within $2 \leq \Delta \gamma_{\omega} \leq 25$.

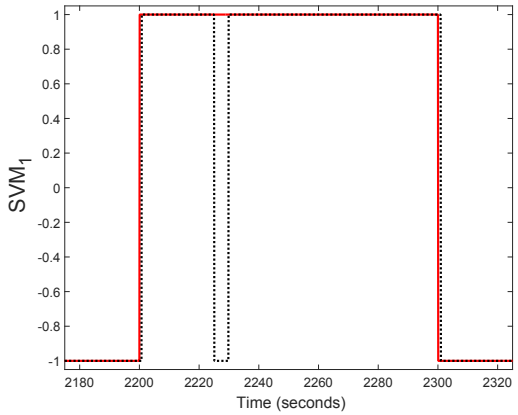
4.2.1.2 Performance Measure

In order to evaluate the performance of the developed FDI scheme, a confusion matrix (also known as the binary contingency table) approach is used as illustrated in Table 4.2. Given a classifier and a data instance, there are four possible outcomes that are [125]:

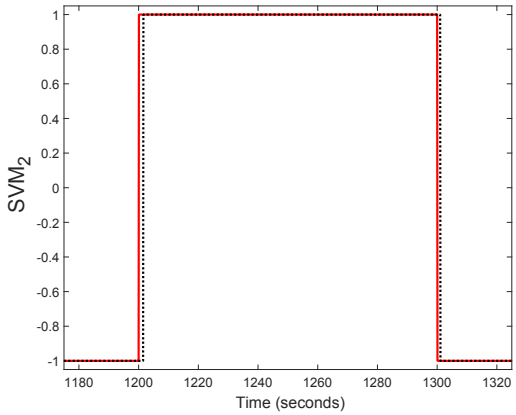
i) a True Positive (TP) if the instance has been *correctly identified*,

ii) a True Negative (TN) if the instance has been *correctly rejected*,

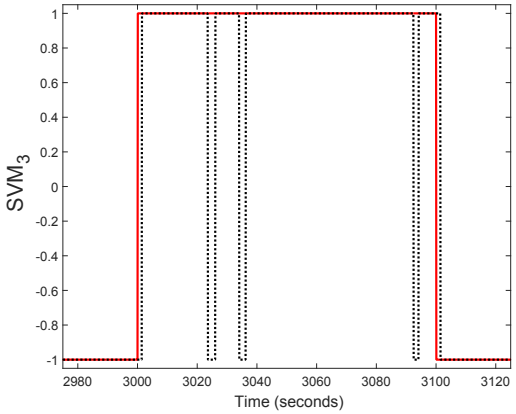
iii) a False Positive (FP) if the instance has been *incorrectly identified*,



(a) The FDI results for the Fault 1 by using the SVM.



(b) The FDI results for the Fault 2 by using the SVM.



(c) The FDI results for the Fault 3 by using the SVM.

Figure 4.7: The FDI results for non-simultaneous faults by using the SVM (dotted line: the output of the SVM, red line: the nominal fault vector).

Table 4.2: Confusion matrix. (Color coding: Green indicates correct counts, whereas red indicates incorrect counts.)

		Predicted	
		Faulty	Healthy
Actual	Faulty	TP	FN
	Healthy	FP	TN

iv) a False Negative (FN) if the instance has been *incorrectly rejected*.

In this thesis, three performance measures based on above possible outcomes are used and they are the Detection Rate (DR), the False Alarm rate (FA), and the F-score ($\mathcal{F}$). These metrics are formally defined as follows:

$$\text{Detection Rate (DR)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.3a}$$

$$\text{False Alarm Rate (FA)} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{4.3b}$$

$$F\text{-score}\,(\mathcal{F}) = \frac{2\text{TP}}{2\text{TP} + \text{FN} + \text{FP}} \tag{4.3c}$$

Table 4.3: Average performance measures for the FDI of the non-simultaneous fault scenarios.

Faults	**Methodology**	**DR%**	**FA%**	**F-score ($\mathcal{F}$)**
Fault 1	SVM_1	93.05%	0.398%	0.8846
	NSA_0 & NSA_1	95.69%	0.624%	0.8595
Fault 2	SVM_2	97.07%	0.383%	0.9081
	NSA_0 & NSA_2	99.06%	0.396%	0.9162
Fault 3	SVM_3	90.46%	0.514%	0.8495
	NSA_0 & NSA_3	88.96%	0.713%	0.8078

4.2.2 Fault Scenario 2: Simultaneous Faults

In this scenario, simultaneous faults (also known as concurrent faults) are considered and injected into the WT in order to evaluate the performance of the second layer of the NSAs. The following three simultaneous fault scenarios are considered:

4.2.2.1 Case 1: Simultaneous Faults 1 and 2

- Fault 1 having a loss of effectiveness of $\delta_{f_{P_g}} = 0.80$ during the time interval 1200s to 1300s.
- Fault 2 having an offset of $\Delta f_\beta = 1.0$ degree during the time interval 1250s to 1350s.

The corresponding WT output measured variables are now plotted in Figure 4.8. The fault detection objective is achieved by utilizing the $\text{NSA}_\mathbf{0}$, where the corresponding fault residual $r_\mathbf{0}(t)$ is shown in Figure 4.9(a). It follows that the anomaly is detected during the time period of 1200s – 1350s. By comparing the $\text{NSA}_\mathbf{0}$ with the $\text{NSA}_\mathbf{1}$, it can be concluded that during the time period 1200s – 1250s, only Fault 1 has occurred. Since the $\text{NSA}_\mathbf{1}$ is sensitive to all faults, but Fault 1, it considers Fault 2 that starts at 1250s as making the WT faulty.

Similar reasoning can be made for Fault 2 by comparing the $\text{NSA}_\mathbf{0}$ and $\text{NSA}_\mathbf{2}$ (as shown in Figure 4.9(a)). The fault residual $r_{\mathbf{1,2}}(t)$ corresponding to the $\text{NSA}_{\mathbf{1,2}}$ is shown in the bottom graph of Figure 4.9(a). The $\text{NSA}_{\mathbf{1,2}}$ filter is activated only when

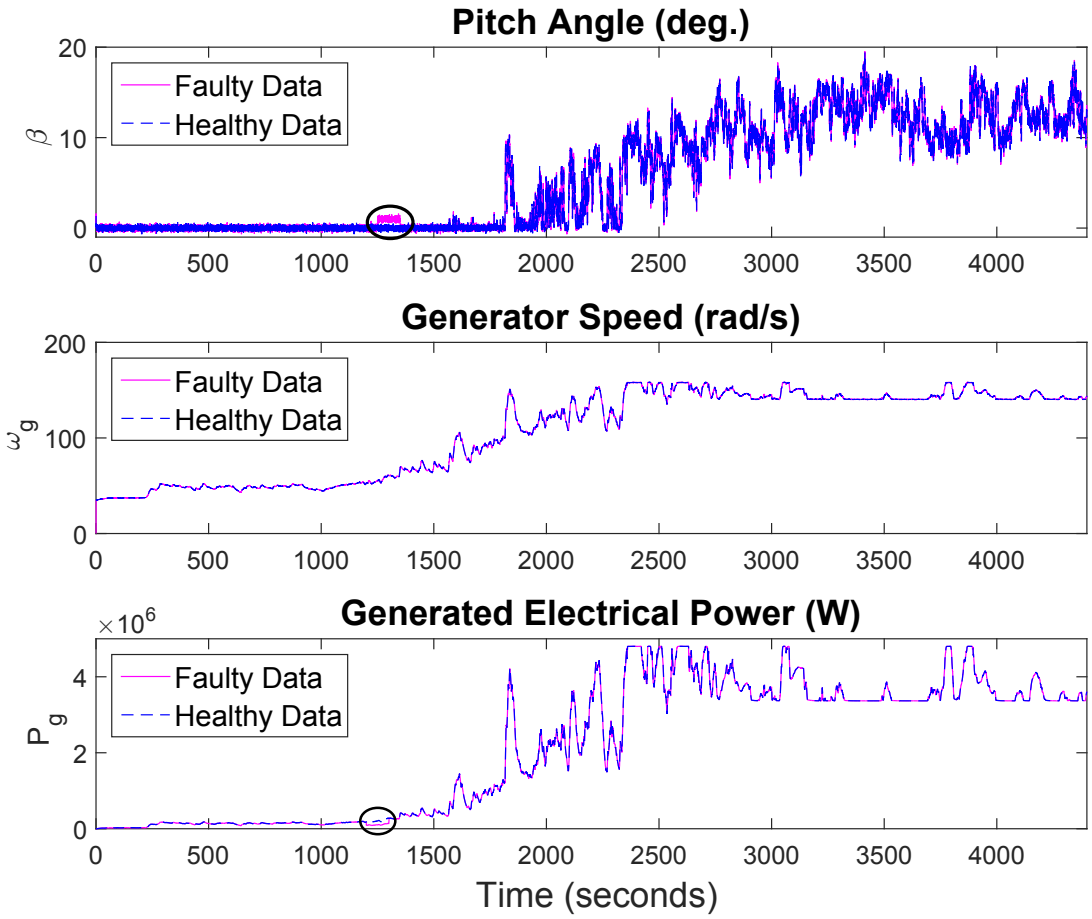


Figure 4.8: The WT measured variables under the healthy and simultaneous Faults 1 and 2 cases.

$r_0(t) = 1$, that is between 1200s – 1350s. Since the $\text{NSA}_{1,2}$ is trained to be insensitive to simultaneous Faults 1 and 2, it will recognize this faulty scenario as healthy during the time period 1250s – 1300s. Note that the faulty samples during the time periods 1200s – 1250s and 1300s – 1350s in $\text{NSA}_{1,2}$ correspond to any fault scenario rather than the presence of concurrent Faults 1 and 2.

Figure 4.10(a) depicts the FDI results for simultaneous Faults 1 and 2 by using both the SVM_1 and SVM_2. The time duration when the output of both SVMs predict the faulty samples corresponds to the case when both Faults 1 and 2 occur simultaneously. As shown in Figure 4.10(a), the SVM_1 and SVM_2 both produce an output of +1 (implying a faulty sample) during 1250s - 1300s. Consequently, one can conclude that the concurrent Faults 1 and 2 have occurred during that time period.

4.2.2.2 Case 2: Simultaneous Faults 1 and 3

- Fault 1 having a loss of effectiveness of $\delta_{f_{Pg}} = 0.85$ during the time interval 3200s to 3300s.
- Fault 3 having a damping factor of $\Delta\gamma_\omega = 15$ during the time interval 3250s to 3350s.

A similar conclusion for the FDI as in Case 1 can be drawn for simultaneous isolation of Faults 1 and 3. The corresponding FDI results for the NSA and SVM schemes are shown in Figures 4.9(b) and 4.10(b), respectively. The $NSA_{1,3}$ which is responsible for isolating the concurrent Faults 1 and 3, is activated only when $r_0(t) = 1$. Therefore, during 3200s – 3350s by comparing the $NSA_{1,3}$ with NSA_0, it can be concluded that during the time period 3250s – 3000s, Faults 1 and 3 have occurred simultaneously. It should be noted that there are few missed detections for NSA_0, NSA_1, NSA_3 as well as few false alarms for the $NSA_{1,3}$.

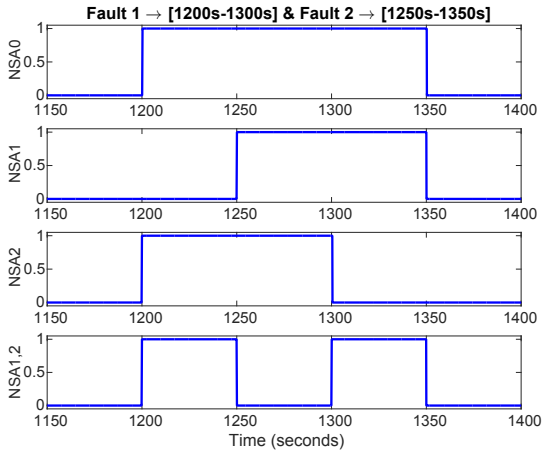
4.2.2.3 Case 3: Simultaneous Faults 2 and 3

- Fault 2 having an offset of $\Delta f_\beta = 1.0$ degree during the time interval 1200s to 1300s.
- Fault 3 having a damping factor of $\Delta \gamma_\omega = 15$ during the time interval 1250s to 1350s.

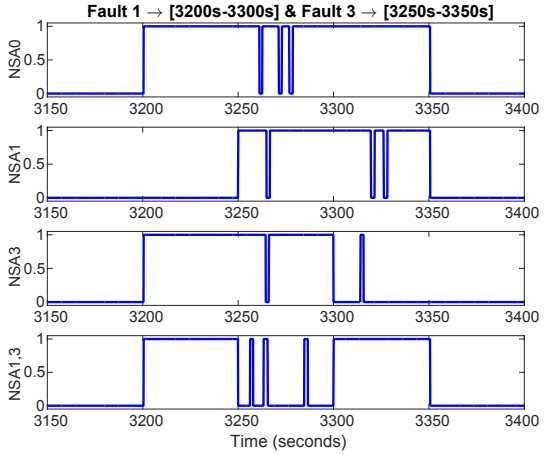
A similar conclusion for the FDI as in Case 1 can be drawn for the simultaneous isolation of Faults 2 and 3. The corresponding FDI results for the NSA and SVM schemes are shown in Figures 4.9(c) and 4.10(c), respectively. In this fault scenario, $\text{NSA}_{2,3}$ (which is responsible for isolating the concurrent Faults 2 and 3) is activated only when $r_0(t) = 1$. Therefore, during 1200s – 1350s by comparing the $\text{NSA}_{2,3}$ with NSA_0, it follows that simultaneous Faults 2 and 3 have occurred during the time period 1250s – 1300s. Note that in this case there are also few missed detections for NSA_0, NSA_2, NSA_3 as well as few false alarms for the $\text{NSA}_{2,3}$.

Table 4.4: Average performance measures for the FDI of the simultaneous faults.

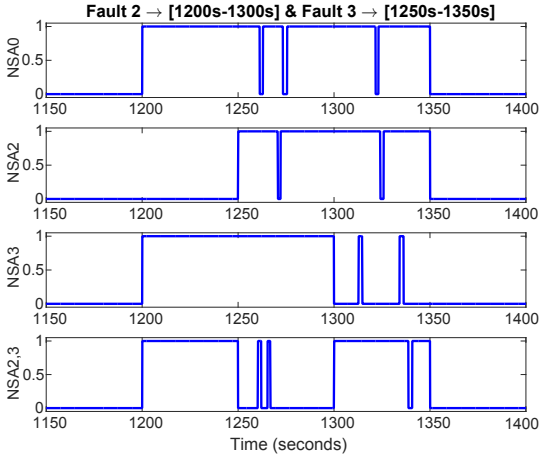
Faults	**Methodology**	**DR%**	**FA%**	***F*-score ($\mathcal{F}$)**
Faults 1 & 2	SVM_1 & SVM_2	93.24%	0.382%	0.8883
	NSA_0 & NSA_{12}	93.70%	0.512%	0.8677
Faults 1 & 3	SVM_1 & SVM_3	92.61%	0.737%	0.8255
	NSA_0 & NSA_{13}	89.73%	1.022%	0.7657
Faults 2 & 3	SVM_2 & SVM_3	94.41%	0.664%	0.8460
	NSA_0 & NSA_{23}	90.98%	0.830%	0.8011



(a) The FDI results for simultaneous Faults 1 and 2 by using the NSA.

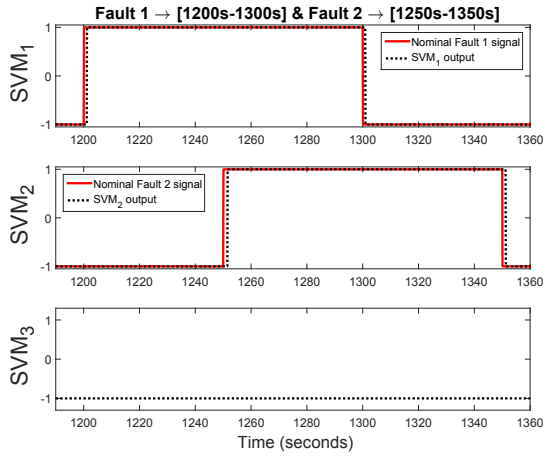


(b) The FDI results for simultaneous Faults 1 and 3 by using the NSA.

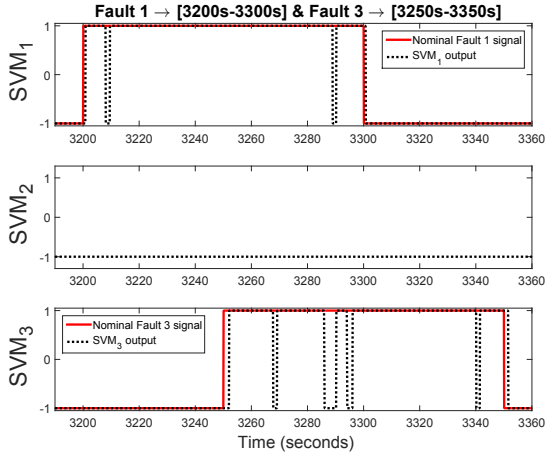


(c) The FDI results for simultaneous Faults 2 and 3 by using the NSA.

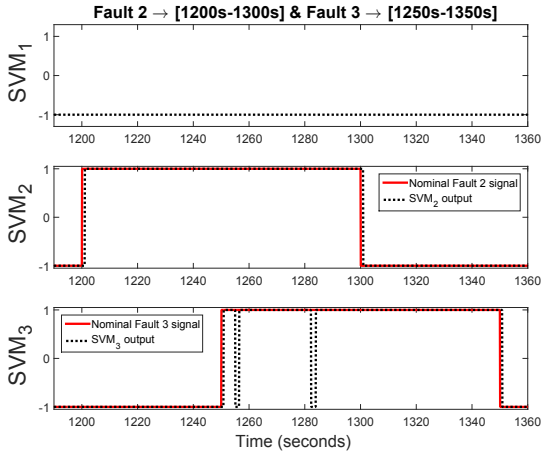
Figure 4.9: The FDI results for simultaneous faults by using the NSA.



(a) The FDI results for simultaneous Faults 1 and 2 by using the SVM.



(b) The FDI results for simultaneous Faults 1 and 3 by using the SVM.



(c) The FDI results for simultaneous Faults 2 and 3 by using the SVM.

Figure 4.10: The FDI results for simultaneous faults by using the SVM (red line: nominal fault vector, dotted line: the output of the SVM).

4.2.3 Non-Parametric Statistical Comparison

In order to compare the NSA-based methodology with the SVM, a *non-parametric statistical* comparison test is implemented. Non-parametric tests can be implemented for two types of analysis, namely: *pairwise comparisons* and *multiple comparisons* [126]. A *hypothesis testing* is employed to draw inferences on the comparison results between the proposed NSA-based methodology and the SVM. In this chapter, the *null hypothesis* ($\boldsymbol{H_0}$) is the statement that there is no difference between the proposed methodology and the SVM, whereas the *alternative hypothesis* ($\boldsymbol{H_1}$) shows that there is a significant difference between the proposed methodology and the SVM. In order to reject a hypothesis, a significance level $\boldsymbol{\alpha}$ is introduced that is the probability below which $\boldsymbol{H_0}$ may be rejected [126].

In this thesis, a pairwise statistical procedure is chosen due to its feature of performing an individual comparison between two algorithms. The number of times that an algorithm wins or loses are counted based on the F-score ($\mathcal{F}$). F-score is used as a single performance metric to determine which algorithm outperforms (the NSA-based methodology or the SVM). The reason for choosing the $\mathcal{F}$ is due to the fact that in addition to considering truly identified samples (TP), it also takes both the falsely rejected (FN) as well as the falsely identified (FP) samples into account.

The number of wins or losses are compared based on the *sign test* (a form of a two-tailed binomial test) [126]. Therefore, in the sign test, the number of wins is distributed

according to a binomial distribution. The metrics Wins(+) and Loses(-) can be formally defined as follows:

$$\text{Wins}(+) = \sum_{k=1}^{NumSim} \mathcal{U}\left(\text{NSA}_{\mathcal{F}}(k) - \text{SVM}_{\mathcal{F}}(k)\right), \tag{4.4}$$

$$\text{Loses}(-) = NumSim - \text{Wins}(+), \tag{4.5}$$

where $NumSim$ denotes the total number of simulation runs, $\text{NSA}_{\mathcal{F}}$ and $\text{SVM}_{\mathcal{F}}$ denote the F-scores of the proposed NSA-based FDI scheme and the SVM, respectively. $\mathcal{U}()$ represents the step function that is defined as follows:

$$\mathcal{U}(x) = \begin{cases} 1 & x > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Since, the proposed NSA-based methodology is tested for 25 different fault severities for each fault scenario (*i.e.*, $NumSim = 25$), the critical number of wins that is needed to achieve the significance levels of $\alpha = 0.05$ or $\alpha = 0.1$ are 18 and 17 wins out of 25, respectively (refer to Table B.1 in Appendix B). More details on the sign test and other non-parametric statistical procedures in both pairwise as well as multiple comparisons are available in [126]. The metrics Wins(+) and Loses(-) are evaluated for each fault scenario considered in Sections 4.2.1 and 4.2.2 and the corresponding non-parametric test results are listed in Table 4.5.

Table 4.5: Sign test for comparing the performance of NSA against SVM (only the number of wins/loses of NSA is indicated).

		Wins (+)	**Loses (−)**	**Difference**
Non–Simultaneous Faults	**Fault 1**	14	11	–
	Fault 2	20	5	$\alpha = 0.05$
	Fault 3	6	19	–
Simultaneous Faults	**Faults 1&2**	12	13	–
	Faults 1&3	6	19	–
	Faults 2&3	8	17	–

4.3 Discussion

As mentioned earlier, for comparing the performance of the proposed NSA scheme with that of the SVM, three quantitative metrics or measures are used, namely: the *Detection Rate*, **DR%**, *False Alarm rate*, **FA%**, and the F-score. The average performance results for both non-simultaneous and simultaneous fault scenarios are shown in Tables 4.3 and 4.4, respectively. The non-parametric statistical comparison results based on the *sign test* are provided in Table 4.5.

In case of the SVM scheme, it follows that the false alarm rates are lower than that of the proposed NSA scheme. This is due to the better generalization performance (error rates on the test data set) of the SVM. As far as the F-score, the SVM algorithm shows better performance in all cases except in case of Fault 2. This is due to the fact that the SVM has lower false alarm rates as compared to the NSA. For example, in case of Fault 1, the average detection rate of NSA is higher and even it outperforms the SVM

(14 wins against 11 wins for the SVM). However, the mean F-score is in favor of the SVM due to the higher false alarm rates of the NSA. Simulation results corresponding to the proposed NSA-based methodology and the SVM show similar results based on the pairwise comparison test (Table 4.5). In some cases, the NSA outperforms the SVM. For instance, in case of Fault 2, the NSA shows a significantly better performance as compared to the SVM with a level of significance $\alpha = 0.05$. However, in case of non-simultaneous Fault 3, the SVM has a better performance as compared to the NSA in isolating the Fault 3.

The most significant limitation of the SVM is in the choice of a proper kernel function. This issue has also been raised in previous works such as [127]. For the SVM scheme, the input data is transformed into a higher dimensional space where they can be separated linearly. The kernel function has a significant role in this mapping process and the selection of the best choice for a given problem is still an open area of research. In this work, a polynomial kernel function is used (equation (4.2)). The use of the polynomial kernel requires tuning several parameters simultaneously. The problem of fine tuning the kernel parameters exists in other kernel functions such as the Gaussian Radial Basis Function (RBF). On the other hand, the main parameter in implementing the V-detector that needs to be tuned is the threshold of data instances (the self-radius of the spheres), which is not a challenging task. Note that the two parameters, namely the maximum number of detectors and the coverage of non-self region are often initialized

at the start of the process and do not require to be fine tuned thereafter. In this work, polynomial kernels of degrees 3 and 5 are used for developing the SVM models, whereas all the NSAs have the same properties as mentioned in Section 4.2.

Another advantage of the V-detector is its less computational complexity during the training phase as compared with the SVM. The complexity of the V-detector algorithm is $O(m|S_{tr}|)$, where m denotes the preset number of detectors and $|S_{tr}|$ denotes the size of the training set [116]. On the other hand, the computational complexity for solving the nonlinear SVM regardless of the particular solver algorithm lies between $O(|S_{tr}|^2)$ and $O(|S_{tr}|^3)$ [128], [129]. In this work, the preset number of detectors was chosen to be 10000 (hence, $m = 10000$), and the size of the training set, $|S_{tr}|$, is selected as 35,200 samples.

4.4 Conclusions

In this chapter, an FDI scheme based on the NSA is developed. The proposed NSA-based FDI scheme is applied to the WT benchmark in order to detect and isolate certain fault scenarios (both simultaneous as well as non-simultaneous faults). The proposed scheme includes a bank of NSAs that are configured into a hierarchical structure for the fault isolation task. Moreover, a moving window filter is utilized to improve the FDI performance. Finally, a non-parametric statistical comparison test is implemented

to compare the performance of the proposed NSA-based FDI methodology with a well-known data-driven technique, namely the Support Vector Machine (SVM).

Chapter 5

DCA-based FDI Scheme

In this chapter, a sensor FDI framework is developed based on Dendritic Cell Algorithm (DCA) to detect and isolate sensor faults within a system. The proposed DCA-based FDI scheme is then compared with the previously developed NSA-based FDI scheme, and a non-parametric statistical comparison test is conducted.

5.1 Proposed DCA-based FDI Scheme

In the proposed DCA-based FDI methodology, it is assumed that sensors are physically redundant in order to be able to detect and isolate the corresponding sensor faults. The justification behind this assumption is that modern industrial WTs employ conservative hardware redundant FDI and health monitoring systems [33, 130, 131]. For instance, the work [132] developed an observer-based fuzzy FTC method for the FTC of Wind Energy

Conversion Systems (WECSs), in which a multi-sensor scheme using hardware redundant sensors is employed. Likewise, the work [133] exploits the physically redundant sensor measurements to generate the necessary fault residuals for the fault detection of the WT benchmark model.

The main steps for implementation of the proposed DCA-based FDI scheme is shown in Figure 5.1.

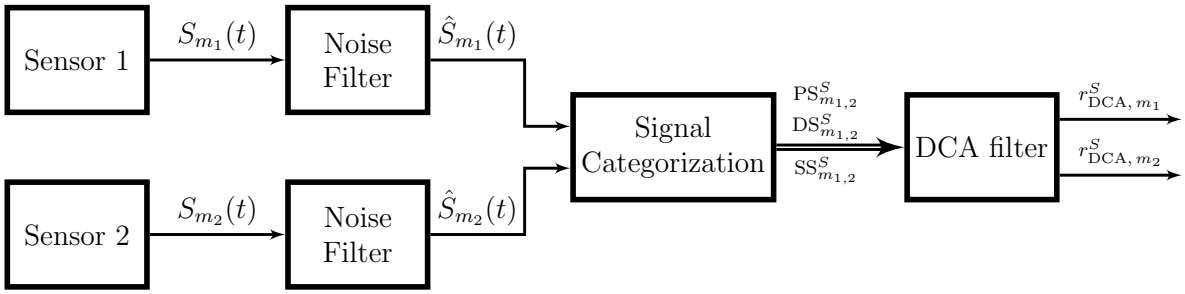


Figure 5.1: The proposed DCA-based sensor FDI scheme.

5.1.1 Noise Filter Design

A design of the noise filter is recommended in order to remove measurement noise. A noise filter processes highly noisy data providing a more suitable data for FDI purposes. A first-order low-pass filter is employed for this purpose. The transfer function of the filter is given as follows:

$$\frac{\hat{S}_{m_j}(s)}{S_{m_j}(s)} = \frac{1}{Ts+1} \tag{5.1}$$

where $\hat{S}_{m_j}(s) = \mathcal{L}\{\hat{s}_{m_j}(t)\}$ and $S_{m_j}(s) = \mathcal{L}\{s_{m_j}(t)\}$ denote the Laplace transforms of the filtered and the raw data, respectively, and T denotes the time constant of the transfer function. A discrete-time realization of equation (5.1) by using backward Euler method is provided below

$$\hat{s}_{m_j}[k] = \alpha\, s_{m_j}[k] + (1-\alpha)\, \hat{s}_{m_j}[k-1], \tag{5.2}$$

where $\hat{s}_{m_j}[k]$ and $s_{m_j}[k]$ denote the filtered and the raw data at time k, respectively, and $\alpha \in (0, 1]$ denotes the filter smoothing factor that can be calculated as follows:

$$\alpha = \frac{\Delta t}{T + \Delta t}$$

where Δt denotes the time step.

The time constant of the filter, T, is chosen such that the following condition, namely the Nyquist criterion, is met [134].

$$\mathrm{B} < \frac{f_s}{2}, \tag{5.3}$$

where $f_s = 1/\Delta t$ denotes the sampling frequency and $\mathrm{B} = 1/(2\pi T)$ denotes the bandwidth of the filter (in Hz).

5.1.2 Signal Categorization

The objective of the *Signal Categorization* block is to prepare the input signals for each sensor (PS, DS, SS) that are applied to the DCA as follows:

- **PAMP Signal (PS):** PS signal is chosen to be the difference between two sensor readings that measure the same output as shown below. The reason behind this selection is due to the big difference between the two sensor readings that measure the same quantity indicates a strong possibility of an abnormality in the system. The PS signal can be expressed as:

$$\mathrm{PS}^{S}_{m_j}(t) = (-1)^{j+1}\left(\hat{S}_{m_1}(t) - \hat{S}_{m_2}(t)\right),$$

- **Danger Signal (DS):** DS signal is chosen to be the difference between two consecutive time samples of a measurement. The DS signal can be expressed as:

$$\mathrm{DS}^{S}_{m_j}(t) = \hat{S}_{m_j}(t) - \hat{S}_{m_j}(t-1).$$

- **Safe Signal (SS):** Similar to the DS, the SS signal is chosen to be the difference between the current time and the previous time samples, namely

$$\mathrm{SS}^{S}_{m_j}(t) = \hat{S}_{m_j}(t) - \hat{S}_{m_j}(t-1).$$

In addition to above three input signals, a non-time varying signal (IC) is used to amplify the effects of the other three signals. Moreover, in this work a binary variable DC is defined for each sensor, which is either 1 or 0 depending on the concentration of above three signals.

5.1.3 DCA Filter

For each sensor, a DCA filter is assigned that generates a binary residual vectors (*fault indicator*) showing the behavior of the corresponding sensor. If the j-th sensor is faulty at time t, then $r^{S}_{\mathrm{DCA},m_j}(t) = 1$. Similarly, $r^{S}_{\mathrm{DCA},m_j}(t) = 0$ corresponds to a normal behavior of the sensor at time t. For instance, in case of the sensor β_{i,m_j}, the fault residual is as follows:

$$r^{\beta_i}_{\mathrm{DCA},m_j}(t) = \begin{cases} 1 & \text{the } j\text{-th sensor measuring } \beta_i \text{ is faulty at time } t \\ 0 & \text{the } j\text{-th sensor measuring } \beta_i \text{ is healthy at time } t \end{cases} \tag{5.4}$$

In order to generate the DCA residual signals (r^{S}_{DCA,m_j}), the output of each DC, which determines the maturity level of the DC, is evaluated based on the concentration of the input signals (PS, DS, and SS) as follows:

$$\mathrm{DC}^{S}_{\mathrm{out},m_j}(t) = \frac{W_{\mathrm{PS}}\,\mathrm{PS}^{S}_{m_j}(t) + W_{\mathrm{DS}}\,\mathrm{DS}^{S}_{m_j}(t) + W_{\mathrm{SS}}\,\mathrm{SS}^{S}_{m_j}(t)}{W_{\mathrm{PS}} + W_{\mathrm{DS}} + W_{\mathrm{SS}}} \cdot \frac{1+\mathrm{IC}}{2} \tag{5.5}$$

where W_{PS}, W_{DS}, and W_{SS} denote the signal weights that are predefined by users and can be either positive or negative, depending on the values of PS, DS, and SS in the immune system. In the original DCA developed by Greensmith [65], the signal weights are chosen based on empirical biological data. If $\mathrm{DC}^{S}_{\mathrm{out},m_j}(t)$ is less than the *Migration Threshold* ($\mathrm{DC}^{S}_{\mathrm{out},m_j}(t) < \textit{Migration Threshold}$), the corresponding DC matures to a smDC (refer to Section 3.2.1), and hence DC = 0. However, if $\mathrm{DC}^{S}_{\mathrm{out},m_j}(t) > \textit{Migration Threshold}$, then the DC matures to a mDC and consequently DC = 1. A new variable $\mathrm{DC}_{\mathrm{store}}$

is defined that stocks previous status of DC for the last *TimeWindow* time samples. The variable *TimeWindow* is crucial for constructing an online FDI decision criterion or logic. The procedure of temporarily storing the status of DC is known as the *Time-Based Segmentation* (TBS), in which a fixed segment size (*TimeWindow*) is adapted within the DCA filter as suggested in [120]. Then, if the number of mDC (Num_mDC) is greater than a given *Threshold*, all content of that *TimeWindow* will be set to 1. Pseudocode 2 illustrates the step by step implementation of the proposed DCA filter. In order to have a fault indicating signal for a sensor at each time sample t, the variable $r^{S}_{\mathrm{DCA},m_j}(t)$, which is a global count, is introduced that returns a binary value at time t.

5.2 Simulation Results

In this section, the sensor faults considered in the WT benchmark model [13] are presented and the corresponding details are provided. Then, the simulation results of the proposed DCA-based FDI scheme and comparison with the proposed NSA-based FDI scheme in Chapter 4 are provided.

5.2.1 Fault Scenarios

Overall, there are five sensor faults that are considered in the WT benchmark model [13] as modeled and described in Section 2.4.1.1. Table 5.1 lists all injected fault scenarios

Pseudocode 2: Pseudocode of the proposed DCA filter.

Input: Antigen = $[\hat{S}_{m_1}(t),\ \hat{S}_{m_2}(t)]$ `// Filtered Sensor Measurements`

Output: $r^{S}_{\mathrm{DCA},m_j}(t)$ $(j \in \{1,2\})$ `// Residual Signals`

Initialization

Number of Antigen = Number of Sensors;

Migration Threshold; `// DC lifespan`

IC; `// inflammatory signal`

$TimeWindow$;

$Threshold$; `// Fault threshold`

foreach *DC* **do**

  Calculate $\mathrm{PS}^{S}_{m_j}(t)$, $\mathrm{DS}^{S}_{m_j}(t)$, $\mathrm{SS}^{S}_{m_j}(t)$;

  Calculate $DC^{S}_{\mathrm{out},m_j}$ from equation (5.5);

  if $DC^{S}_{out,m_j} <$ *Migration Threshold* **then**

    $DC = 0$; `// iDC` $\rightarrow$ `smDC`

  else

    $DC = 1$; `// iDC` $\rightarrow$ `mDC`

  end

  $\mathrm{DC}_{\mathrm{store}}(t - TimeWindow) = DC$;

  Num_mDC = 0; `// # of mDC in` $\mathrm{DC}_{\mathrm{store}}$

  for *all members of* DC_{store} **do**

    if $DC_{store} == 1$ **then**

      Num_mDC++;

    end

  end

  if *Num_mDC* $>$ *Threshold* **then** Set all members

    $\mathrm{DC}_{\mathrm{store}} = 1$;

  end

  for *each Antigen type* **do**

    $r^{S}_{\mathrm{DCA},m_j}(t) = \mathrm{DC}_{\mathrm{store}}(t)$;

  end

end

Table 5.1: All sensor faults considered in the benchmark model [13].

Fault #	Faulty Sensor	Fault Type	Fault Severity	Injected Time
S1	Blade 1 pitch position (β_{1,m_1})	Fixed value	$\Delta\beta_{1,m_1} = 5°$	2000s to 2100s
S2	Blade 2 pitch position (β_{2,m_2})	Gain factor	$\Delta\beta_{2,m_2} = 1.2$	3200s to 3300s
S3	Blade 3 pitch position (β_{3,m_1})	Fixed value	$\Delta\beta_{3,m_1} = 10°$	2600s to 2700s
S4	Rotor speed sensor (ω_{r,m_1})	Fixed value	$\Delta\omega_{r,m_1} = 1.4$ rad/s	1500s to 1600s
S5	Generator and rotor speed sensors (ω_{r,m_2} & ω_{g,m_2})	Gain factor	$\Delta\omega_{r,m_2} = 1.1$ $\Delta\omega_{g,m_2} = 0.9$	1000s to 1100s

that are considered in the WT model. More detail about each fault type is available in Subsection 2.4.1.1.

5.2.2 Simulation Results of Proposed DCA-based FDI

A distributed version of the proposed DCA-based FDI scheme for sensor faults (Figure 5.1) is applied to the WT system and is illustrated in Figure 5.2, where the sensor FDI of each WT subsystem is performed apart from the other subsystems. The corresponding DCA fault residuals for each sensor are shown in Figures 5.4 – 5.9.

5.2.2.1 Noise Filter Parameters

A low-pass filter is employed to reduce the noisy pitch position as well as the rotational speed measurements (refer to Section 5.1.1 for the design procedure). The time constants of the filters (T in equation (5.1)) are selected as follows: $T_\beta = 0.06$s for the noise filter designed for the pitch position sensor measurement, $T_{\omega_r} = 0.75$s and $T_{\omega_g} = 0.02$s

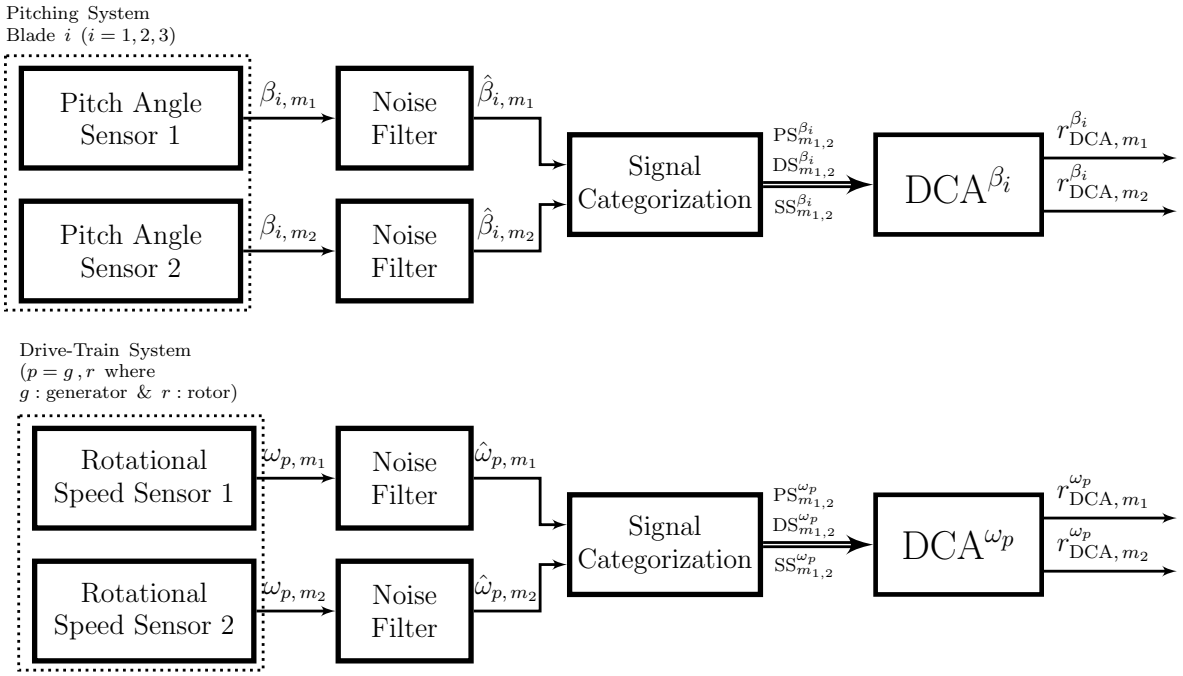
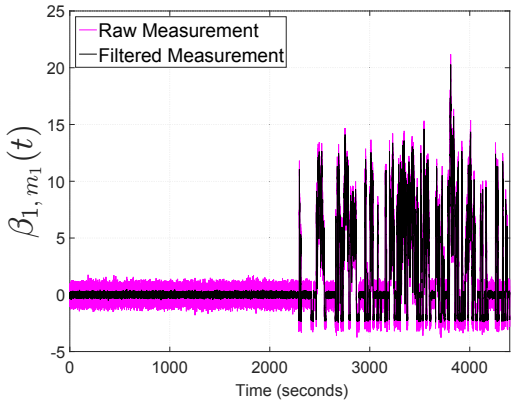


Figure 5.2: The proposed DCA-based sensor FDI scheme for the benchmark WT model.

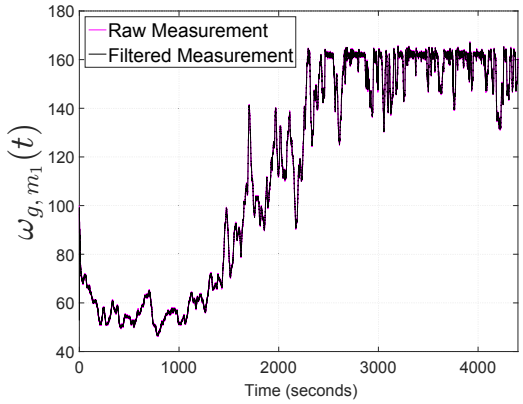
for the rotor and generator speed measurements. It should be noted that the time constants meet the Nyquist criterion (equation (5.3)). However, they are tuned empirically based on the filtering performance. Figure 5.3 illustrates the raw and the filtered sensor measurements using the above selected time constants. Note that the gain factor fault in rotor speed measurements, ω_{r,m_j}, *cannot* be observed if the noise filter is not implemented.

5.2.2.2 DCA Filter Parameters

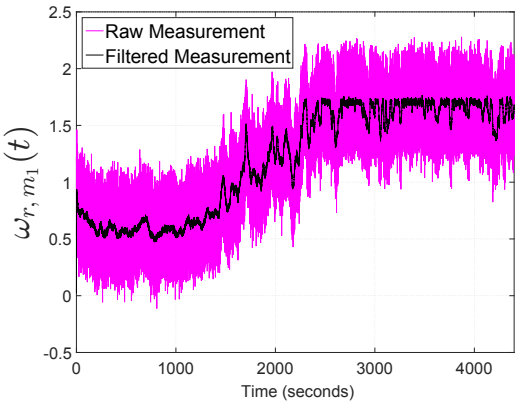
The parameters of the DCA filter are user-specified and are selected based on empirical results obtained from implementations. Given the dual sensor redundancy, the number of



(a) Low-pass filter applied to the $\beta_m(t)$ with $T_\beta = 0.06$s.



(b) Low-pass filter applied to the $\omega_g(t)$ with $T_{\omega_g} = 0.02$s.



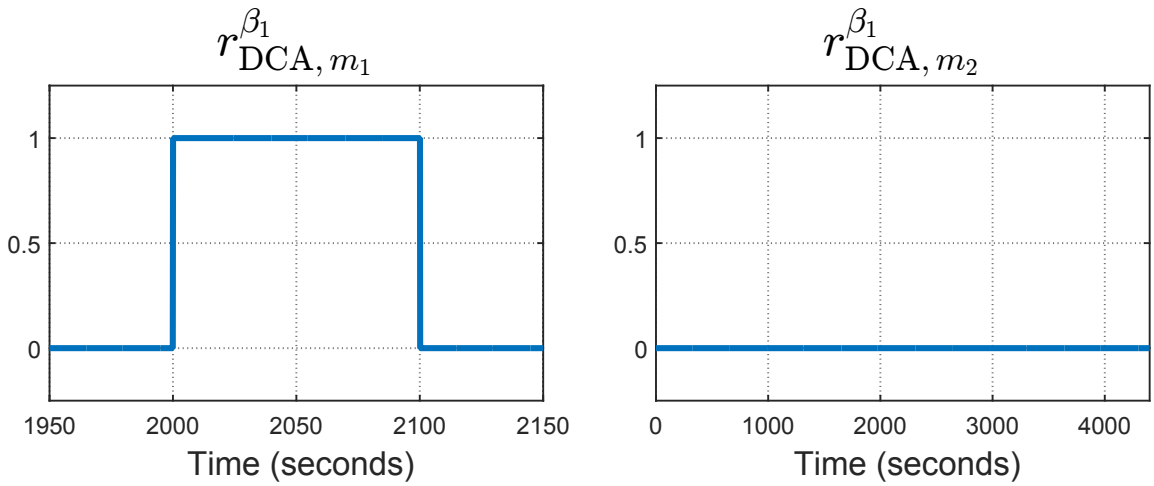
(c) Low-pass filter applied to the $\omega_r(t)$ with $T_{\omega_r} = 0.75$s.

Figure 5.3: Raw and filtered sensor measurements for $\beta_{i,m_j}(t)$, $\omega_g(t)$, and $\omega_r(t)$.

Antigens is chosen to be two. The constant signal IC in equation (5.5) that amplifies the other input signals is trivial in most cases (*i.e.*, IC = 1) except in case of the FDI of ω_{r,m_j} sensor that is selected as IC = 27. The reason behind this selection is due to the difficulty in FDI of the gain factor fault in rotor speed measurements, ω_{r,m_j}, and the need for amplification of the other input signals ($\text{PS}^{\omega_r}_{m_j}$, $\text{DS}^{\omega_r}_{m_j}$, and $\text{SS}^{\omega_r}_{m_j}$). The signals weights in equation (5.5) are chosen based on the number of sensors (in our case two sensors

for each subsystem) as follows: $W_{\mathrm{PS}} = -2$, $W_{\mathrm{DS}} = 2$, and $W_{\mathrm{SS}} = -2$. Moreover, the other DCA filter parameters are empirically chosen as follows: *Migration Threshold* = 1 , $TimeWindow = 10$, and $Threshold = 8$.

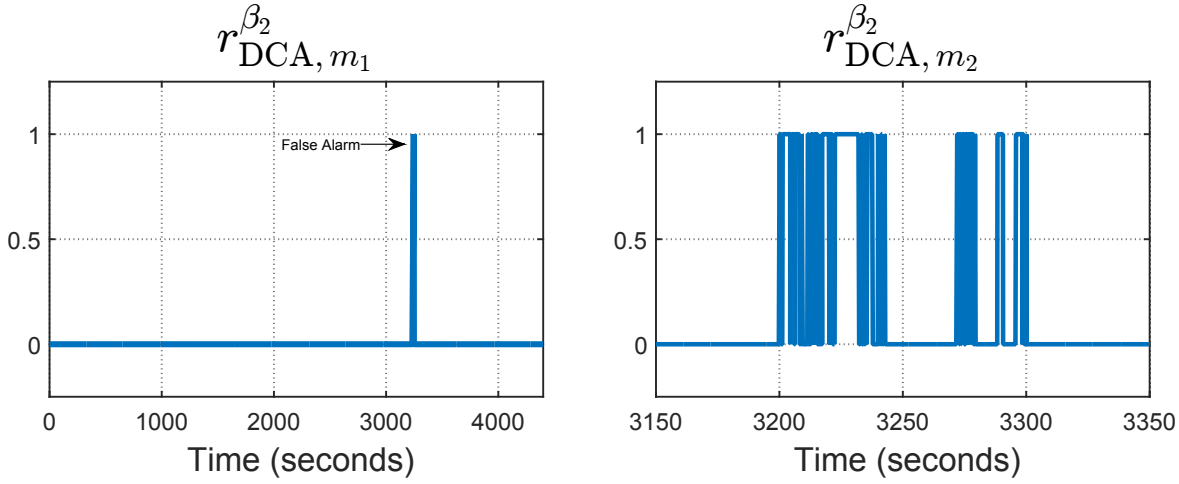
The DCA filters are capable of detecting and isolating all the injected sensor faults with few false alarms. Figures 5.4 – 5.9 show the residuals generated by implementing the DCA-based FDI scheme (Figure 5.2). As can be seen in Figure 5.4(a), the residual signal $r^{\beta_1}_{\mathrm{DCA},m_1} = 1$ during 2000 – 2100s meaning that the blade 1 pitch position sensor β_{1,m_1} is faulty during that time period. Similarly, faults in sensors ω_{r,m_1} (1500 – 1600s) and ω_{g,m_2} (1000 - 1100s) are detected and isolated since the residual signals $r^{\omega_r}_{\mathrm{DCA},m_2} = 1$ and $r^{\omega_g}_{\mathrm{DCA},m_2} = 1$ in Figures 5.7(a) and 5.9(b), respectively. There are few false alarms in case of sensors β_{2,m_1} and β_{3,m_2} in which DCA^{β_2} and DCA^{β_3} incorrectly output $r^{\beta_2}_{\mathrm{DCA},m_1} = 1$ and $r^{\beta_3}_{\mathrm{DCA},m_2} = 1$ as indicated in Figures 5.5(a) and 5.6(b), respectively. In case of Fault S2 in the sensor β_{2,m_2} that is of the type gain factor (refer to Table 5.1), not all the samples during 3200 - 3300s are considered to be faulty as illustrated in Figure 5.5(b). This is due to the fact that whenever the pitch position is at zero angle ($\beta_{2,m_2} = 0°$), the faulty sensor shows $\Delta\beta_{2,m_2} = 1.2 \times 0° = 0°$, and hence no fault can be detected. Similarly, in case of Fault S5 in the sensor ω_{r,m_2}, the $\mathrm{DCA}^{\omega_r}_{m_2}$ has missed detecting and isolating few faulty samples during 1000 - 1100s as shown in Figure 5.8(b).



(a) $r^{\beta_1}_{\mathrm{DCA},m_1}$ residual signal.

(b) $r^{\beta_1}_{\mathrm{DCA},m_2}$ residual signal.

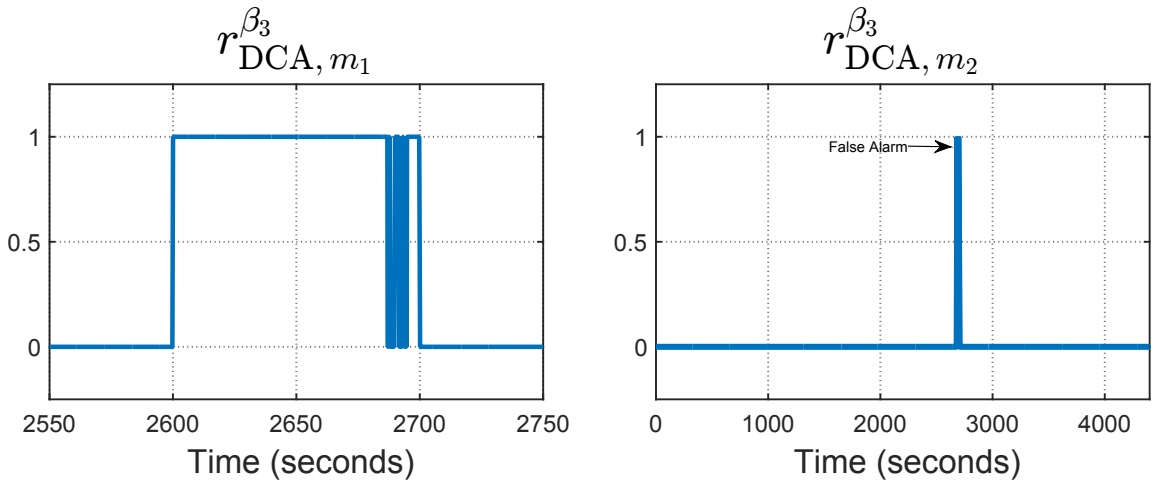
Figure 5.4: The residual signal due to the Fault S1. The DCA$^{\beta_1}$ residual signals for the pitch sensors of the blade 1.



(a) $r^{\beta_2}_{\mathrm{DCA},m_1}$ residual signal.

(b) $r^{\beta_2}_{\mathrm{DCA},m_2}$ residual signal.

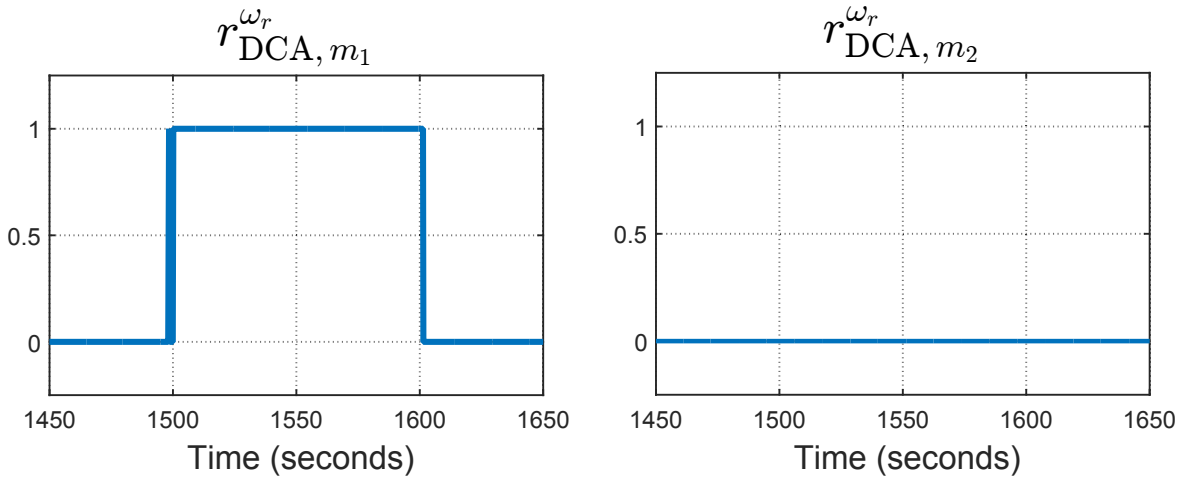
Figure 5.5: The residual signal due to the Fault S2. The DCA$^{\beta_2}$ residual signals for the pitch sensors of the blade 2.



(a) $r^{\beta_3}_{\mathrm{DCA},m_1}$ residual signal.

(b) $r^{\beta_3}_{\mathrm{DCA},m_2}$ residual signal.

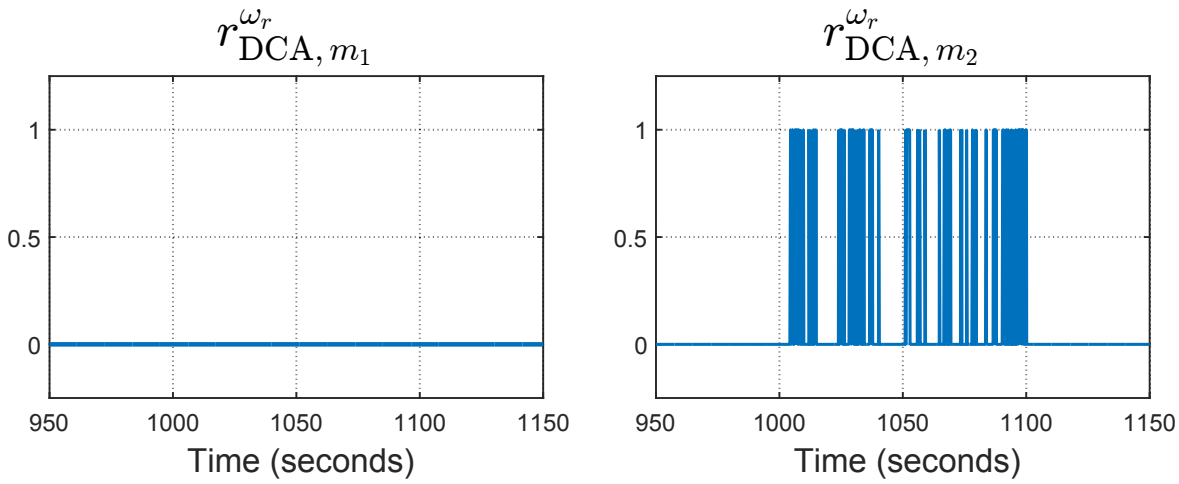
Figure 5.6: The residual signal due to the Fault S3. The DCA$^{\beta_3}$ residual signals for the pitch sensors of the blade 3.



(a) $r^{\omega_r}_{\mathrm{DCA},m_1}$ residual signal.

(b) $r^{\omega_r}_{\mathrm{DCA},m_2}$ residual signal.

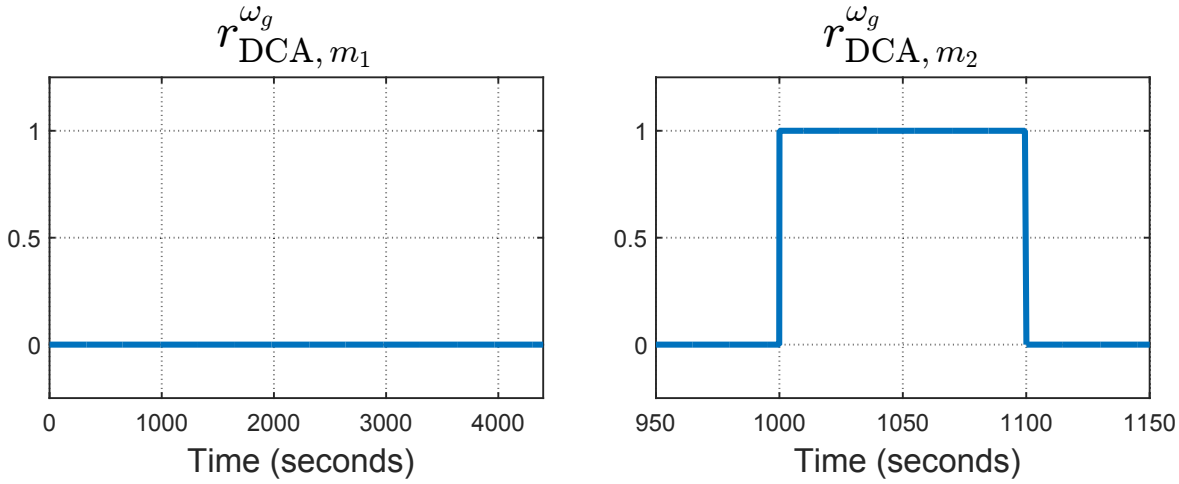
Figure 5.7: The residual signal due to the Faults S4. The DCA$^{\omega_r}$ residual signals for the rotor speed sensors.



(a) $r^{\omega_r}_{\mathrm{DCA},m_1}$ residual signal.

(b) $r^{\omega_r}_{\mathrm{DCA},m_2}$ residual signal.

Figure 5.8: The residual signal due to the Faults S5. The DCA$^{\omega_r}$ residual signals for the rotor speed sensors.



(a) $r^{\omega_g}_{\mathrm{DCA},m_1}$ residual signal.

(b) $r^{\omega_g}_{\mathrm{DCA},m_2}$ residual signal.

Figure 5.9: The residual signal due to the Fault S5. The DCA$^{\omega_g}$ residual signals for the generator speed sensors.

5.2.3 Comparison with the Proposed NSA-based FDI

In this section, simulation results from previous section are compared with the proposed NSA-based methodology developed in Chapter 4.

Figure 5.10 shows the bank of NSAs used that is used for the task of FDI. As noted in Figure 5.10, the features determining the shape space are chosen to be the filtered sensor measurements of each subsystem, as in Figure 5.2. By choosing similar inputs to the proposed DCA-based methodology, a fair comparison can be made in evaluating the performance of each framework.

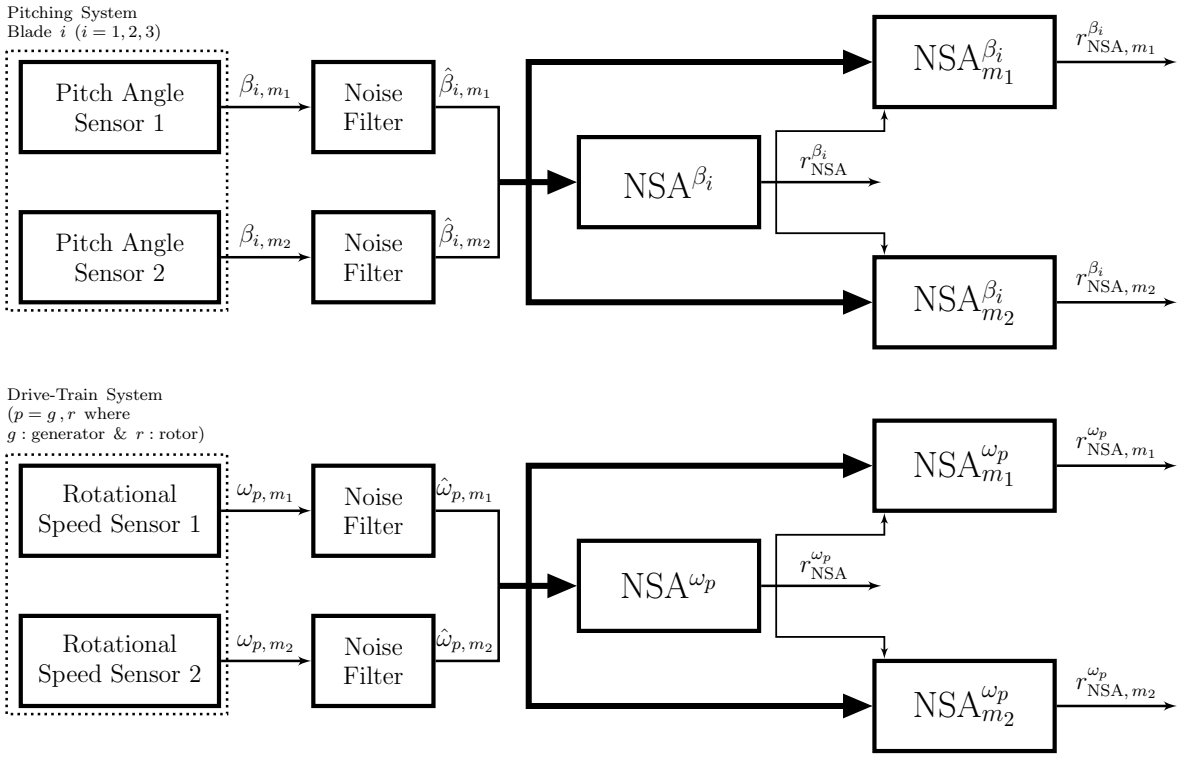


Figure 5.10: The NSA-based sensor FDI scheme for the benchmark WT model.

A similar procedure explained in Chapter 4 is applied here to perform the sensor

FDI in the WT system. For performing the fault detection task, the first NSA, namely NSA^{β_i} is trained to detect any change in the behavior of the pitch actuator i and to remain insensitive to only the healthy data samples generated by the pitch actuator system. The residual signal generated by NSA^{β_i} is designated by $r_{\text{NSA}}^{\beta_i}$. However, for isolating the fault, two NSAs are constructed where each NSA is trained to be insensitive to a particular actuator. For instance, $\text{NSA}_{m_j}^{\beta_i}$ is trained to be insensitive to the blade i pitch sensor j. A similar fault detection as well as fault isolation procedure are applied to the rotational speed sensors. It should be noted that 80% of the data is allocated for the training stage while the remaining 20% is allocated for the test stage.

In order to compare DCA and NSA frameworks (Figures 5.2 and 5.10, respectively), 25 Monte Carlo simulation runs are conducted for each fault type corresponding to randomly selected fault severities. Table 5.2 lists the average performance for the FDI of the sensor faults. Moreover, a nonparametric statistical comparison test is conducted based on the results of the Monte Carlo simulation runs, in which the *null hypothesis* ($\boldsymbol{H_0}$) is the statement that there is no difference between the DCA-based FDI scheme and the NSA-based FDI scheme, whereas the *alternative hypothesis* ($\boldsymbol{H_1}$) shows that there is a significant difference between both schemes.

The result of the nonparametric statistical test is listed in Table 5.3, in which the number of wins refer to the number of simulation cases (out of 25) that the DCA-based FDI scheme outperforms the NSA-based FDI one. Similar to equation (4.4), the metric

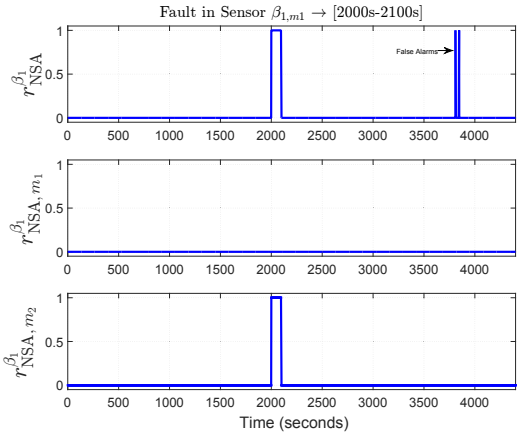
Wins(+) can be formally defined as follows:

$$\text{Wins}(+) = \sum_{k=1}^{NumSim} \mathcal{U}\left(\text{DCA}_{\mathcal{F}}^{S}(k) - \text{NSA}_{\mathcal{F}}^{S}(k)\right), \tag{5.6}$$

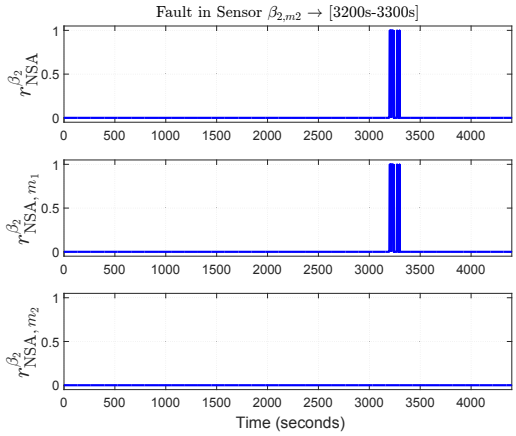
where $\text{DCA}_{\mathcal{F}}^{S}$ and $\text{NSA}_{\mathcal{F}}^{S}$ denote the F-scores of the DCA-based and the NSA-based FDI schemes for the sensor $S(t)$, respectively. Like in previous chapter, $NumSim = 25$ and Loses(-) can be evaluated according to equation (4.5).

Table 5.2: Average performance measures for the FDI of the sensor faults.

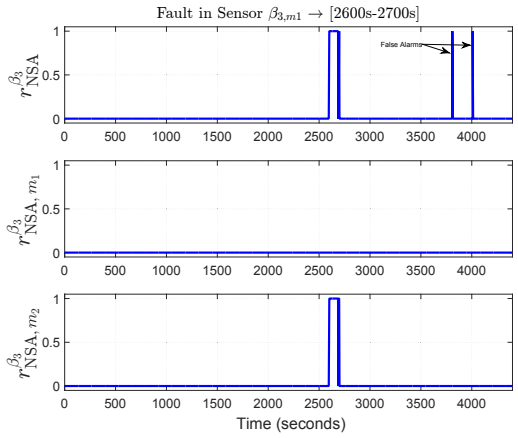
Faults	**Methodology**	**DR%**	**FA%**	**F-score ($\mathcal{F}$)**
Fault S1	DCA^{β_1}	98.06%	0.061%	0.9769
	NSA^{β_1} & $\text{NSA}_{m_1}^{\beta_1}$	97.49%	0.220%	0.9417
Fault S2	DCA^{β_2}	–	–	–
	NSA^{β_2} & $\text{NSA}_{m_2}^{\beta_2}$	–	–	–
Fault S3	DCA^{β_3}	97.42%	0.139%	0.9575
	NSA^{β_3} & $\text{NSA}_{m_1}^{\beta_3}$	97.96%	0.157%	0.9565
Fault S4	DCA^{ω_r}	97.88%	0.135%	0.9607
	NSA^{ω_r} & $\text{NSA}_{m_1}^{\omega_r}$	98.15%	0.204%	0.9481
Fault S5	DCA^{ω_g}	98.57%	0.073%	0.9771
	NSA^{ω_g} & $\text{NSA}_{m_2}^{\omega_g}$	98.42%	0.119%	0.9705
	DCA^{ω_r}	79.19%	0.355%	0.8115
	NSA^{ω_r} & $\text{NSA}_{m_2}^{\omega_r}$	–	–	–



(a) NSA residual signals generated for the pitch sensors of balde 1.

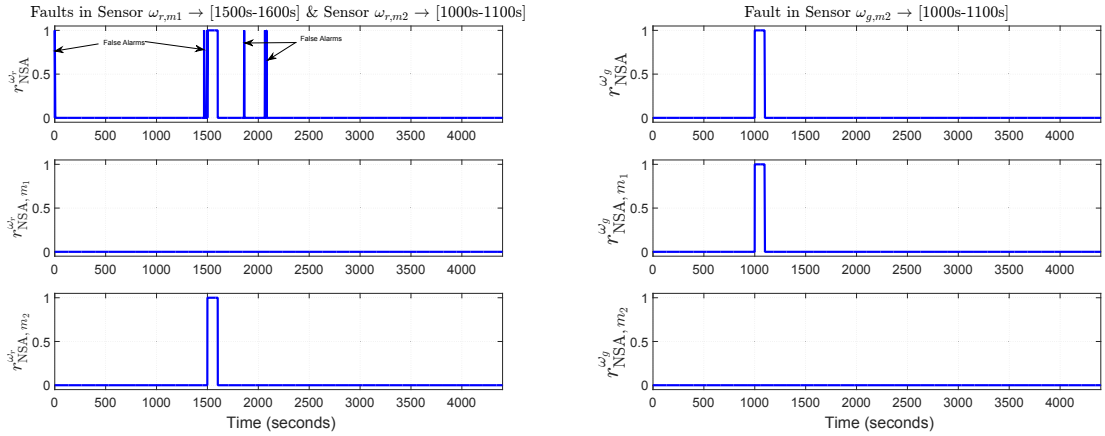


(b) NSA residual signals generated for the pitch sensors of balde 2.



(c) NSA residual signals generated for the pitch sensors of balde 3.

Figure 5.11: The NSA residual signals for the WT pitch sensors.



(a) NSA residual signals generated for the rotor speed sensors.

(b) NSA residual signals generated for the generator speed sensors.

Figure 5.12: The NSA residual signals for rotor and generator speed sensors.

5.3 Discussion

As shown in Table 5.2, the DCA has a higher F-score in all cases. This is due to the higher false alarm rates of the NSA-based FDI scheme as compared to the DCA-based FDI scheme. Even in the case that NSA-based FDI scheme has shown to have, on average, better detection rates (namely in Faults S3 and S4), the DCA-based FDI scheme has a higher F-score.

In case of Fault S2, the performance cannot be fairly measured due to the fact that the pitch position *can* be at zero angle, $\beta_i(t) = 0$, (see Section 2.3), and hence $\Delta f_{\beta_{i,m_j}} = 0$ no matter what the value of $\Delta\beta_{i,m_j}$ is (refer to equation (2.25)). In case of Fault S5 that affects both the rotor as well as generator speed sensors simultaneously,

NSA-based FDI scheme could not detect and isolate the sensor fault in ω_{r,m_2} during $1000\text{s} - 1100\text{s}$.

Table 5.3 illustrates results of the nonparametric comparison analysis between the DCA-based and NSA-based FDI schemes. The number of Wins(+) and Loses(-) of the DCA-based FDI scheme is considered here (refer to equations (5.6) and (4.5), respectively). As can be seen, the DCA-based FDI scheme outperforms the NSA-based FDI scheme with the significance level of $\alpha = 0.1$ in FDI of all sensor faults except Fault S3. The main reason for better results in case of the DCA-based FDI scheme is due to high false alarm rates of the NSA. The issue of high false alarm has also been raised in Chapter 4. Since no performance measure is obtained in case of Fault S2, hence the sign test cannot be conducted for this fault.

Table 5.3: Sign test for comparing the performance of DCA against NSA (only the number of wins/loses of DCA is indicated).

	Wins (+)	**Loses (−)**	**Difference**
Fault S1	22	3	$\alpha = 0.1$
Fault S2	–	–	–
Fault S3	10	15	–
Fault S4	19	6	$\alpha = 0.1$
Fault S5	20	5	$\alpha = 0.1$
	25	0	$\alpha = 0.1$

According to [120], the runtime complexity of the standard DCA is bounded by $O(n^2)$, where n denotes the data size, and the runtime complexity of the DCA with

segmentation is bounded by $O(\max(nN, nz))$, where N denotes the DC population size, and z denotes the segment size ($TimeWindow$). On the other hand, the complexity of V-detector is $O(mn)$ (Section 4.3), where m denotes the preset number of detectors. Hence, both the V-detector as well as the DCA version (integrated with Time-Based Segmentation (TBS)) used in this thesis have linear runtime complexities. However, DCA is slightly faster than V-detector and that is due to the fact that in this work, the DC population size is $N = 2$ and the segment size $z = 10$, whereas the preset number of detectors in the V-detector algorithm is chosen to be 1000, and hence $m = 1000$. Moreover, the developed DCA-based FDI scheme does not require a training phase while the NSA-based FDI scheme requires a detector generation (training) phase.

A common FDI approach in the literature in case of dual sensor redundancy is to first detect the fault by a direct comparison of the dual sensors, and then to perform the isolation task using one of analytical redundancy approaches. However, the proposed DCA-based FDI scheme performs both fault detection as well as fault isolation tasks, simultaneously.

5.4 Conclusions

In this chapter, an FDI scheme based on the DCA is developed. The proposed DCA-based FDI scheme is applied to the WT benchmark in order to detect and isolate all sensor faults considered in the benchmark model. A time-segmentation is integrated

within the DCA filter in order to perform the FDI task in an online manner and also to reduce the runtime complexity. Finally, a non-parametric statistical comparison test is implemented to compare the performance of the proposed DCA-based FDI scheme with the previously developed NSA-based FDI scheme in Chapter 4.

Chapter 6

Conclusions and Future Work

In this thesis, the problem of Fault Detection and Isolation (FDI) of the wind turbine system is addressed by using two different immune inspired algorithm, namely the Negative Selection Algorithm (NSA) and the Dendritic Cell Algorithm (DCA).

First, a bank of NSAs is proposed and implemented for detecting and isolating faults in the WT. The fault detection objective is achieved by training an NSA with only normal or healthy data. However, both healthy and faulty data are used for accomplishing the fault isolation task and various NSAs are configured into a hierarchical structure that are trained to isolate different fault scenarios, including concurrent fault cases. To improve the FDI performance, as well as to remove the outliers, a moving window NSAs is applied to the residual signals that are generated by the NSAs. Multiple fault scenarios are presented to highlight the performance capabilities of the

proposed FDI scheme. The results are also compared with the Support Vector Machine (SVM) scheme, where the detection and isolation performance versus the computational complexity and simplicity in training are all taken into account and evaluated. A non-parametric statistical comparison test is also implemented to compare the proposed NSA-based FDI methodology with the SVM corresponding to various fault severities. The SVM and NSA simulation results show a similar performance, and under certain fault scenarios, the NSA outperforms the SVM.

The second scheme that is developed in this thesis is an DCA-based online sensor FDI methodology. The time segmentation (time window) is integrated within the DCA filters to develop an online FDI methodology as well as to reduce the runtime complexity. The main utility and advantage of the proposed DCA-based methodology is that no training data is needed, which is an important feature of DCA. The proposed framework is tested on the WT benchmark model in which the proposed scheme is successful in detecting and isolating all sensor fault scenarios that are introduced in the benchmark. Moreover, the proposed DCA-based FDI methodology is compared with the NSA-based FDI scheme. Monte Carlo simulation runs as well as non-parametric statistical comparison test are conducted to illustrate the effectiveness and capabilities of the DCA-based proposed scheme.

The NSAs as well as DCA, and in general immune system inspired schemes are still a young research field. More studies should be conducted to further develop these

techniques in the field of FDI of complex industrial systems.

6.1 Future Work

Some suggestions for future work on both NSA and the DCA are provided below:

6.1.1 Future Work on the NSA

- Optimizing the detector generation mechanism of the NSA with the objective of covering as much of the non-self space as possible with a minimum number of detectors (and hence reducing detectors' overlap).
- Investigating the issue of high false alarm rates (due to the high False Positive (FP)) of the NSA, and consequently improving its performance.
- Conducting studies on the relationship between data instances, and consequently developing a population-based version of the NSA (rather than an instance-based algorithm).
- Conducting more research on the use of new metrics and distances to be used in the detector generation as well as the monitoring phase of the NSA algorithm, especially in the case of high-dimensional data.
- Addressing the issue of the input scaling as a necessary preprocessing step of the NSA, particularly in case of real-valued NSA.

- Conducting studies on the use of different detector shapes (in addition to the hypersphere) in case of real-valued NSA, or even to use various detector shapes.

- Integrating an online learning mechanism in to the NSA for updating detectors in the detector set.

- Automatic tuning of NSA parameters such as the self-radius of data instances in case of real-valued NSA.

6.1.2 Future Work on the DCA

- Automating the data pre-processing phase and exploring the use of feature selection techniques to choose suitable input signals for the DCA.

- Developing a framework for the FDI of actuator faults as well as system faults in theWT benchmark model. A pre-processing mechanism to choose input signals of the DCA may be necessary for this task.

- Developing a systematic way of choosing weights for signal transformation with the goal of maximizing the detection accuracy.

- Integrating a dynamic segmentation into the DCA.

- Automatic tuning of DCA parameter, particularly in case of the original DCA that includes many stochastic parameters.

Bibliography

[1] R. Isermann, *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer Science & Business Media, 2006.

[2] S. Simani, C. Fantuzzi, and R. J. Patton, *Model-based Fault Diagnosis in Dynamic Systems Using Identification Techniques*. Springer, 2003.

[3] S. X. Ding, *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*, 2nd ed. Springer London, 2013.

[4] I. Hwang, S. Kim, Y. Kim, and C. E. Seah, "A Survey of Fault Detection, Isolation, and Reconfiguration Methods," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 3, pp. 636–653, May 2010.

[5] M. N. Desai, J. C. Deckert, and J. J. Deyst Jr., "Dual-Sensor Failure Identification Using Analytic Redundancy," *Journal of Guidance, Control, and Dynamics*, vol. 2, no. 3, pp. 213–220, 1979.

[6] E. P. Kim and N. R. Shanbhag, "Soft N-Modular Redundancy," *IEEE Transactions on Computers*, vol. 61, no. 3, pp. 323–336, March 2012.

[7] S. Osder, "Practical View of Redundancy Management Application and Theory," *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 1, pp. 12–21, 1999.

[8] J. Zhang, A. K. Swain, and S. K. Nguang, *Robust Observer-Based Fault Diagnosis for Nonlinear Systems Using MATLAB®*. Springer, 2016.

[9] Z. Gao, C. Cecati, and S. X. Ding, "A Survey of Fault Diagnosis and Fault-Tolerant Techniques – Part I: Fault Diagnosis With Model-Based and Signal-Based Approaches," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3757–3767, June 2015.

[10] ——, "A Survey of Fault Diagnosis and Fault-Tolerant Techniques – Part II: Fault Diagnosis With Knowledge-Based and Hybrid/Active Approaches," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3768–3774, June 2015.

[11] N. Meskin and K. Khorasani, *Fault Detection and Isolation: Multi-Vehicle Unmanned Systems*. Springer Science & Business Media, 2011.

[12] P. F. Odgaard, J. Stoustrup, and M. Kinnaert, "Fault Tolerant Control of Wind Turbines a benchmark model," in *7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, vol. 42, no. 8, 30 June – 3 July 2009, pp. 155 – 160.

[13] ——, "Fault-Tolerant Control of Wind Turbines: A Benchmark Model," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1168–1182, 2013.

[14] H. Sanchez, T. Escobet, V. Puig, and P. Odgaard, "Fault Diagnosis of an Advanced Wind Turbine Benchmark Using Interval-Based ARRs and Observers," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3783–3793, June 2015.

[15] P. Casau, P. Rosa, S. M. Tabatabaeipour, C. Silvestre, and J. Stoustrup, "A Set-Valued Approach to FDI and FTC of Wind Turbines," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 245–263, January 2015.

[16] W. Chen, S. X. Ding, A. Haghani, A. Naik, A. Q. Khan, and S. Yin, "Observer-based FDI Schemes for Wind Turbine Benchmark," in *Proceedings of the 18th IFAC World Congress*, vol. 2, 2011, pp. 7073–7078.

[17] P. Pisu and B. Ayalew, "Robust Fault Diagnosis for a Horizontal Axis Wind Turbine," in *Proceedings of 18th IFAC World Congress*, 2011, pp. 7055–7060.

[18] J. Blesa, D. Rotondo, V. Puig, and F. Nejjari, "FDI and FTC of wind turbines using the interval observer approach and virtual actuators/sensors," *Control Engineering Practice*, vol. 24, pp. 138 – 155, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S096706611300230X

[19] X. Zhang, Q. Zhang, S. Zhao, R. M. Ferrari, M. M. Polycarpou, and T. Parisini, "Fault Detection and Isolation of the Wind Turbine Benchmark: An Estimation-based Approach," in *Proceedings of the 18th IFAC World Congress*, vol. 2, 2011, pp. 8295–8300.

[20] F. Stoican, C.-F. Raduinea, and S. Olaru, "Adaptation of set theoretic methods to the fault detection of wind turbine benchmark," in *Proceedings of the 18th IFAC World Congress*, 2011, pp. 8322–8327.

[21] J. Blesa, V. Puig, J. Romera, and J. Saludes, "Fault Diagnosis of Wind Turbines using a Set-membership Approach," in *Proceedings of IFAC world congress*, 2011, pp. 8316–8321.

[22] S. Yin, G. Wang, and H. R. Karimi, "Data-driven design of robust fault detection system for wind turbines," *Mechatronics*, vol. 24, no. 4, pp. 298 – 306, 2014.

[23] I. V. de Bessa, R. M. Palhares, M. F. S. V. D'Angelo, and J. E. C. Filho, "Data-driven fault detection and isolation scheme for a wind turbine benchmark," *Renewable Energy*, vol. 87, Part 1, pp. 634 – 645, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0960148115304146

[24] R. M. Fernandez-Canti, J. Blesa, S. Tornil-Sin, and V. Puig, "Fault detection and isolation for a wind turbine benchmark using a mixed bayesian/set-membership approach," *Annual Reviews in Control*, vol. 40, pp. 59 – 69, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578815000395

[25] S. Simani, S. Farsoni, and P. Castaldi, "Fault Diagnosis of a Wind Turbine Benchmark via Identified Fuzzy Models," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3775–3782, June 2015.

[26] S. Simani and P. Castaldi, "Data-Drive Design of Fuzzy Logic Fault Tolerant Control for a Wind Turbine Benchmark," in *8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SAFEPROCESS)*, vol. 45, no. 20, Aug. 29–31, 2012, pp. 108–113. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016347395

[27] S. Simani, P. Castaldi, and A. Tilli, "Data-Driven Approach for Wind Turbine Actuator and Sensor Fault Detection and Isolation," in *Proceedings of the 18th IFAC World Congress*, 2011, pp. 8301–8306.

[28] N. Laouti, S. Othman, M. Alamir, and N. Sheibat-Othman, "Combination of Model-Based Observer and Support Vector Machines for Fault Detection of Wind Turbines," *International Journal of Automation and Computing*, vol. 11, no. 3, pp. 274–287, 2014.

[29] N. Sheibat-Othman, S. Othman, M. Benlahrache, and P. F. Odgaard, "Fault detection and isolation in wind turbines using support vector machines and observers," in *American Control Conference (ACC)*. IEEE, Jun. 17–19, 2013, pp. 4459–4464.

[30] J. Zeng, D. Lu, Y. Zhao, Z. Zhang, W. Qiao, and X. Gong, "Wind Turbine Fault Detection and Isolation Using Support Vector Machine and a Residual-Based Method," in *American Control Conference (ACC)*. IEEE, June 2013, pp. 3661–3666.

[31] P. F. Odgaard and J. Stoustrup, "Frequency Based Fault Detection in Wind Turbines," in *the 19th IFAC World Congress*, vol. 19, no. 1, 2014, pp. 5832–5837.

[32] D. Maquin, P. F. Odgaard, H. Sanchez, T. Escobet, and V. Puig, "Fault Diagnosis and Fault Tolerant Control with Application on a Wind Turbine Low Speed Shaft Encoder," *9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2015*, vol. 48, no. 21, pp. 1357 – 1362, Sep. 2 – 4, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896315018431

[33] P. F. Odgaard and J. Stoustrup, "Results of a Wind Turbine FDI Competition," in *8th IFAC symposium on Fault Detection, Supervision and Safety of Technical Processes*, 2012, pp. 102–107.

[34] S. Dey, P. Pisu, and B. Ayalew, "A Comparative Study of Three Fault Diagnosis Schemes for Wind Turbines," *IEEE Transactions on Control Systems Technology*, vol. PP, no. 99, pp. 1–1, 2015.

[35] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, "Self-Nonself Disrimination in a Computer," in *Proceedings of IEEE Symposium on Research in Security and Privacy*, Oakland, CA, May 1994, pp. 202–212.

[36] R. S. Khairy, R. Ghazali, and A. Lasisi, *Real-Valued Negative Selection Algorithms: Ensuring Data Integrity Through Anomaly Detection*. Cham: Springer International Publishing, 2016, pp. 23–32.

[37] N. Ling and F. G. Feng, "Real-valued Dual Negative Selection Technique for Intrusion Detection," *International Journal of Security and Its Applications*, vol. 9, no. 4, pp. 279–288, 2015.

[38] A. S. A. Aziz, A. T. Azar, A. E. Hassanien, and S. E.-O. Hanafy, "Negative Selection Approach Application in Network Intrusion Detection Systems," *arXiv preprint arXiv:1403.2716*, 2014.

[39] J. Kim and P. Bentley, "Negative Selection and Niching by an Artificial Immune System for Network Intrusion Detection," in *Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference*, 1999, pp. 149–158.

[40] Z. Lu, G. Pei, B. Liu, and Z. Liu, "Hardware Implementation of Negative Selection Algorithm for Malware Detection," in *IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC)*, Jun. 1–4, 2015, pp. 301–304.

[41] S. Maiti, C. Garai, and R. Dasgupta, "A Detection Mechanism of DoS Attack using Adaptive NSA Algorithm in Cloud Environment," in *International Conference on Computing, Communication and Security (ICCCS)*. IEEE, 2015, pp. 1–7.

[42] R. Rizwan, F. A. Khan, H. Abbas, and S. H. Chauhdary, "Anomaly Detection in Wireless Sensor Networks Using Immune-Based Bioinspired Mechanism," *International Journal of Distributed Sensor Networks*, 2015.

[43] F. Barani and M. Abadi, "BeeID: Intrusion Detection in AODV-based MANETs Using Artificial Bee Colony and Negative Selection Algorithms," *The ISC International Journal of Information Security*, vol. 4, no. 1, pp. 25–39, 2012.

[44] N. Ling, F. Gao-feng, P. Hai-yun, K. Lee, C. Park, H.-D. Yang, Z. Nan, J.-x. Qu, G.-y. Zhang, X.-z. Wang *et al.*, "Technique for Intrusion Detection based on Minkowsky Distance Negative Selection Algorithm," *International Journal of Security and Its Applications*, vol. 9, no. 12, pp. 1–10, 2015.

[45] L. Zhou, Z. Dai, Y. Dai, and L. Zhao, "Motor Fault Detection and Diagnosis Based on Negative Selection Algorithm," in *5th International Conference on Information Engineering for Mechanics and Materials (ICIMM)*, 2015.

[46] X. Z. Gao, X. Wang, and K. Zenger, "Motor fault diagnosis using negative selection algorithm," *Neural Computing and Applications*, vol. 25, no. 1, pp. 55–65, 2014.

[47] D. Dasgupta, K. KrishnaKumar, D. Wong, and M. Berry, "Negative selection algorithm for aircraft fault detection," in *3rd International Conference on Artificial Immune Systems (ICARIS)*. Springer Berlin Heidelberg, Sep. 13–16, 2004, pp. 1–13.

[48] E. Alizadeh, N. Meskin, M. Benammar, and K. Khorasani, "Fault Detection and Isolation of the Wind Turbine based on the Real-valued Negative Selection Algorithm," in *7th IEEE GCC Conference and Exhibition (GCC)*, Nov. 17–20, 2013, pp. 11–16.

[49] C. C. Coello and N. C. Cortés, "A Parallel Implementation of an Artificial Immune Yystem to Handle Constraints in Genetic Algorithms: Preliminary Results," in *Proceedings of the World on Congress on Computational Intelligence*, vol. 1, May 12 – 17, 2002, pp. 819–824.

[50] S. Fouladvand, A. Osareh, and B. Shadgar, "Distribution Estimation Based Negative Selection Algorithm (DENSA)," in *International Workshop on Artificial Immune Systems (AIS)*. IEEE, 2015, pp. 1–7.

[51] X. Xiao, T. Li, and R. Zhang, "An immune optimization based real-valued negative selection algorithm," *Applied Intelligence*, vol. 42, no. 2, pp. 289–302, 2015. [Online]. Available: http://dx.doi.org/10.1007/s10489-014-0599-9

[52] Z. Ji *et al.*, "A Boundary-Aware Negative Selection Algorithm," in *Proceedings of the 9th International Conference on Artificial Intelligence and Soft Computing, ACTA Press*, 2005.

[53] D. Li, S. Liu, and H. Zhang, "A boundary-fixed negative selection algorithm with online adaptive learning under small samples for anomaly detection," *Engineering Applications of Artificial Intelligence*, vol. 50, pp. 93 – 105, 2016.

[54] ——, "Negative selection algorithm with constant detectors for anomaly detection," *Applied Soft Computing*, vol. 36, pp. 618 – 632, 2015.

[55] C. Palanisamy, M. T. Kumaresan, and S. Varalakshmi, "Combined techniques for detecting email spam using negative selection and particle swarm optimization," *International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)*, no. 3, pp. 1102 – 1106, 2016.

[56] I. Idris, A. Selamat, N. T. Nguyen, S. Omatu, O. Krejcar, K. Kuca, and M. Penhaker, "A combined negative selection algorithm–particle swarm optimization for an email spam detection system," *Engineering Applications of Artificial Intelligence*, vol. 39, pp. 33–44, 2015.

[57] I. Idris, A. Selamat, and S. Omatu, "Hybrid email spam detection model with negative selection algorithm and differential evolution," *Engineering Applications of Artificial Intelligence*, vol. 28, pp. 97 – 110, 2014.

[58] Q. Zhang, A. Qin, L. Shu, G. Sun, and L. Shao, "Vibration sensor based intelligent fault diagnosis system for large machine unit in petrochemical industries," *International Journal of Distributed Sensor Networks*, vol. 2015, pp. 1 – 13, Jan. 2015.

[59] D. W. Taylor and D. W. Corne, "An investigation of the negative selection algorithm for fault detection in refrigeration systems," in *International Conference on Artificial Immune Systems*. Springer, 2003, pp. 34–45.

[60] Y. Tao, M. Hu, and Y. Yu, "A novel negative selection algorithm for recognition problems," *International Journal of Hybrid Information Technology*, vol. 8, no. 11, pp. 101–112, 2015.

[61] F. P. A. Lima, C. R. Minussi, R. B. Bessa, and J. N. Fidalgo, "A modified negative selection algorithm applied in the diagnosis of voltage disturbances in distribution electrical systems," in *18th International Conference on Intelligent System Application to Power Systems (ISAP)*, Sep. 11–16, 2015, pp. 1–6.

[62] A. Lasisi, R. Ghazali, and T. Herawan, "Chapter 11 - application of real-valued negative selection algorithm to improve medical diagnosis," in *Applied Computing in Medicine and Health*, ser. Emerging Topics in Computer Science and Applied Computing, D. Al-Jumeily, A. Hussain, C. Mallucci, and C. Oliver, Eds. Boston: Elsevier, 2016, pp. 231 – 243. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128034682000114

[63] L. Cui, D. Pi, and C. Chen, "BIORV-NSA: Bidirectional inhibition optimization r-variable negative selection algorithm and its application," *Applied Soft Computing*, vol. 32, pp. 544 – 552, 2015.

[64] T. Wen, A. Xu, and J. Tang, "Study on extension negative selection algorithm," *International Journal of High Performance Computing and Networking*, vol. 9, no. 1-2, pp. 1–7, 2016.

[65] J. Greensmith, "The Dendritic Cell Algorithm," Ph.D. dissertation, the University of Nottingham, 2007.

[66] U. Aickelin, P. Bentley, S. Cayzer, J. Kim, and J. McLeod, "Danger theory: The link between AIS and IDS?" in *Artificial Immune Systems*. Springer, 2003, pp. 147–155.

[67] J. Greensmith and U. Aickelin, "The Deterministic Dendritic Cell Algorithm," in *Artificial Immune Systems*. Springer, 2008, pp. 291 – 302.

[68] A. A. Al-Hasan and E.-S. M. El-Alfy, "Dendritic Cell Algorithm for Mobile Phone Spam Filtering ," *Procedia Computer Science*, vol. 52, pp. 244–251, 2015.

[69] D. V. Ng and J.-I. G. Hwang, "Android malware detection using the dendritic cell algorithm," in *International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 1. IEEE, 2014, pp. 257–262.

[70] D. Cui, Q. Zhang, J. Xiong, Q. Li, and M. Liu, "Fault Diagnosis Research of Rotating Machinery Based on Dendritic Cell Algorithm," in *International Conference on Information and Automation*. IEEE, 2015, pp. 1020–1025.

[71] N. Mukhtar, G. M. Coghill, and W. Pang, "FdDCA: A Novel Fuzzy Deterministic Dendritic Cell Algorithm," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '16 Companion. New York, NY, USA: ACM, Jul. 20 – 24, 2016, pp. 1007–1010. [Online]. Available: http://doi.acm.org/10.1145/2908961.2931662

[72] J. L. Amaral, "Fault detection in analog circuits using a fuzzy dendritic cell algorithm," in *Artificial Immune Systems*. Springer, 2011, pp. 294–307.

[73] E. Bendiab and M. K. Kholladi, "Recognition of plant leaves using the dendritic cell algorithm," *International Journal of Digital Information and Wireless Communications (IJDIWC)*, vol. 1, no. 1, pp. 284–292, 2011.

[74] M. Mokhtar, R. Bi, J. Timmis, and A. M. Tyrrell, "A Modified Dendritic Cell Algorithm for On-line Error Detection in Robotic Systems," in *Congress on Evolutionary Computation*. IEEE, 2009, pp. 2055–2062.

[75] R. Bi, J. Timmis, and A. Tyrrell, "The Diagnostic Dendritic Cell Algorithm for Robotic Systems," in *Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–8.

[76] O. Ogundipe, J. Greensmith, G. Roberts *et al.*, "Multipath detection using the dendritic cell algorithm," in *XXIV FIG International Congress*, 2010, pp. 1–14.

[77] R. Oates, J. Greensmith, U. Aickelin, J. Garibaldi, and G. Kendall, "The Application of a Dendritic Cell Algorithm to a Robotic Classifier," in *Artificial Immune Systems*. Springer, 2007, pp. 204–215.

[78] F. Gu, J. Greensmith, R. Oates, and U. Aickelin, "Pca 4 dca: The application of principal component analysis to the dendritic cell algorithm," in *9th Annual Workshop on Computational Intelligence (UKCI 2009)*, vol. 7, Sep. 7–9, 2009, p. 9.

[79] Z. Chelly and Z. Elouedi, "From the General to the Specific: Inducing a Novel Dendritic Cell Algorithm from a Detailed State-of-the-Art Review," *International Journal of Pattern Recognition and Artificial Intelligence*, 2016.

[80] ——, *RST-DCA: A Dendritic Cell Algorithm Based on Rough Set Theory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 480–487. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-34487-9_58

[81] ——, *RC-DCA: A New Feature Selection and Signal Categorization Technique for the Dendritic Cell Algorithm Based on Rough Set Theory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 152–165. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33757-4_12

[82] ——, *QR-DCA: A New Rough Data Pre-processing Approach for the Dendritic Cell Algorithm*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 140–150. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-37213-1_15

[83] ——, *A Fuzzy-Rough Data Pre-processing Approach for the Dendritic Cell Classifier*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 109–120. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-39091-3_10

[84] ——, "A New Data Pre-processing Approach for the Dendritic Cell Algorithm Based on Fuzzy Rough Set Theory," in *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, ser. GECCO '13 Companion. New York, NY, USA: ACM, Jul. 6 – 10, 2013, pp. 163–164. [Online]. Available: http://doi.acm.org/10.1145/2464576.2464657

[85] ——, *A New Hybrid Fuzzy-Rough Dendritic Cell Immune Classifier*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 514–521. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38703-6_60

[86] ——, *Further Exploration of the Fuzzy Dendritic Cell Method*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 419–432. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-22371-6_36

[87] ——, *Supporting Fuzzy-Rough Sets in the Dendritic Cell Algorithm Data Pre-processing Phase*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 164–171. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-42042-9_21

[88] D. Al Azzawi, M. G. Perhinschi, H. Moncayo, and A. Perez, "A dendritic cell mechanism for detection, identification, and evaluation of aircraft failures," *Control Engineering Practice*, vol. 41, pp. 134–148, 2015.

[89] D. A. Azzawi, M. G. Perhinschi, and H. Moncayo, "Artificial Dendritic Cell Mechanism for Aircraft Immunity-Based Failure Detection and Identification," *Journal of Aerospace Information Systems*, vol. 11, no. 7, pp. 467–481, 2014.

[90] E.-S. M. El-Alfy and A. A. AlHasan, "Spam filtering framework for multimodal mobile communication based on dendritic cell algorithm," *Future Generation Computer Systems*, vol. 64, pp. 98 – 107, 2016.

[91] I. Munteanu, A. I. Bratcu, N.-A. Cutululis, and E. Ceanga, *Optimal Control of Wind Energy Systems: Towards a Global Approach*. Springer Science & Business Media, 2008.

[92] D. Dolan and P. Lehn, "Simulation Model of Wind Turbine 3p Torque Oscillations due to Wind Shear and Tower Shadow," in *Power Systems Conference and Exposition, 2006. PSCE '06. 2006 IEEE PES*, Oct 2006, pp. 2050–2057.

[93] T. Esbensen and C. Sloth, "Fault Diagnosis and Fault-Tolerant Control of Wind Turbines," Master's thesis, Aalborg University, Denmark, 2009.

[94] P. A. Lynn, *Onshore and Offshore Wind Energy: An Introduction*. John Wiley & Sons, 2011.

[95] F. D. Bianchi, H. De Battista, and R. J. Mantz, *Wind Turbine Control Systems: Principles, Modelling and Gain Scheduling Design*. Springer Science & Business Media, 2006.

[96] N. Orlando, M. Liserre, R. Mastromauro, and A. Dell'Aquila, "A Survey of Control Issues in PMSG-Based Small Wind-Turbine Systems," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1211–1221, Aug 2013.

[97] R. Pena, J. Clare, and G. Asher, "Doubly fed induction generator using back-to-back pwm converters and its application to variable-speed wind-energy generation," *IEEE Proceedings-Electric Power Applications*, vol. 143, no. 3, pp. 231–241, 1996.

[98] K. E. Johnson, L. Y. Pao, M. J. Balas, and L. J. Fingersh, "Control of variable-speed wind turbines: standard and adaptive techniques for maximizing energy capture," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 70–81, 2006.

[99] P. F. Odgaard and J. Stoustrup, "Fault Tolerant Wind Farm Control – A Benchmark Model," in *IEEE International Conference on Control Applications (CCA)*, 2013, pp. 412–417.

[100] B. Dolan, "Wind Turbine Modelling, Control and Fault Detection," Master's thesis, Technical University of Denmark, DTU, DK-2800 Kgs. Lyngby, Denmark, 2010.

[101] J. E. Hunt and D. E. Cooke, "An adaptive, distributed learning system based on the immune system," in *IEEE International Conference on Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century.*, vol. 3. IEEE, 1995, pp. 2494–2499.

[102] W. Zhang, G. Yen, and Z. He, "Constrained Optimization Via Artificial Immune System," *IEEE Transactions on Cybernetics*, vol. 44, no. 2, pp. 185–198, Feb 2014.

[103] L. N. De Castro and F. J. Von Zuben, "The clonal selection algorithm with engineering applications," in *Proceedings of GECCO*, vol. 2000, 2000, pp. 36–39.

[104] J. Greensmith and U. Aickelin, "Artificial Dendritic Cells: Multi-faceted Perspectives," in *Human-Centric Information Processing Through Granular Modelling*. Springer, 2009, pp. 375–395.

[105] R. A. Goldsby, T. J. Kindt, B. A. Osborne, and J. Kuby, *Immunology*, 5th ed. W. H. Freeman and Company, 2003.

[106] D. Dasgupta and S. Forrest, "An Anomaly Detection Algorithm Inspired by the Immune System," in *Artificial immune systems and their applications*. Springer, 1999, pp. 262–277.

[107] Z. Ji and D. Dasgupta, "V-detector: An efficient negative selection algorithm with probably adequate detector coverage," *Information Sciences*, vol. 179, no. 10, pp. 1390–1406, 2009.

[108] D. Dasgupta and F. González, "An Immunity-Based Technique to Characterize Intrusions in Computer Networks," *IEEE Transactions on Evolutionary Computation,*, vol. 6, no. 3, pp. 281–291, June 2002.

[109] J. M. Shapiro, G. B. Lamont, and G. L. Peterson, "An Evolutionary Algorithm to Generate Hyper-Ellipsoid Detectors for Negative Selection," in *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*. ACM, 2005, pp. 337–344.

[110] S. Balachandran, D. Dasgupta, F. Nino, and D. Garrett, "A Framework for Evolving Multi-Shaped Detectors in Negative Selection," in *Proceedings of the 2007*

IEEE Symposium on Foundations of Computational Intelligence (FOCI 2007), 2007, pp. 401–408.

[111] P. K. Harmer, P. D. Williams, G. H. Gunsch, and G. B. Lamont, "An Artificial Immune System Architecture for Computer Security Applications," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 252–280, Jun 2002.

[112] M. Goswami and A. Bhattacharjee, "Detector Generation Algorithm for Self-Nonself Detection in Artificial Immune System," in *International Conference for Convergence of Technology (I2CT)*, Apr. 6–8, 2014, pp. 1–6.

[113] D. Dasgupta, "Advances in Artificial Immune Systems," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 40–49, 2006.

[114] Z. Ji and D. Dasgupta, "Applicability issues of the real-valued negative selection algorithms," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*. ACM, 2006, pp. 111–118.

[115] J. Zhou, "Negative Selection Algorithms: From the Thymus to V-detector," Ph.D. dissertation, Ph. D. dissertation, The University of Memphis, 2006.

[116] Z. Ji and D. Dasgupta, "Real-valued negative selection algorithm with variable-sized detectors," in *Genetic and Evolutionary Computation–GECCO 2004*. Springer, 2004, pp. 287–298.

[117] P. Matzinger, "Tolerance, danger, and the extended family," *Annual review of immunology*, vol. 12, no. 1, pp. 991–1045, 1994.

[118] Z. Chelly and Z. Elouedi, "A survey of the dendritic cell algorithm," *Knowledge and Information Systems*, pp. 1–31, 2015.

[119] R. Oates, G. Kendall, and J. M. Garibaldi, "Frequency analysis for dendritic cell population tuning," *Evolutionary Intelligence*, vol. 1, no. 2, pp. 145–157, 2008.

[120] F. Gu, J. Greensmith, and U. Aickelin, "Theoretical formulation and analysis of the deterministic dendritic cell algorithm," *Biosystems*, vol. 111, no. 2, pp. 127–135, 2013.

[121] Y. Al-Hammadi, U. Aickelin, and J. Greensmith, "DCA for Bot Detection," in *Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1807–1816.

[122] B. Svetozarevic, P. M. Esfahani, M. Kamgarpour, and J. Lygeros, "A Robust Fault Detection and Isolation Filter for a Horizontal Axis Variable Speed Wind Turbine," in *American Control Conference (ACC),*. IEEE, 2013, pp. 4453–4458.

[123] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[124] Y. Tian, Z. Qi, X. Ju, Y. Shi, and X. Liu, "Nonparallel Support Vector Machines for Pattern Classification," *IEEE Transactions on Cybernetics*, vol. 44, no. 7, pp. 1067–1079, July 2014.

[125] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861 – 874, 2006.

[126] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[127] C. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[128] L. Bottou and C.-J. Lin, "Support Vector Machine Solvers," in *Large Scale Kernel Machines*, L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Eds. MIT Press, 2007, pp. 1–27.

[129] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast Kernel Classifiers with Online and Active Learning," *The Journal of Machine Learning Research*, vol. 6, pp. 1579–1619, Dec. 2005.

[130] K. E. Johnson and P. A. Fleming, "Development, implementation, and testing of fault detection strategies on the national wind technology centers controls advanced research turbines," *Mechatronics*, vol. 21, no. 4, pp. 728 – 736, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957415810002102

[131] H. Badihi, Y. Zhang, and H. Hong, "Fuzzy gain-scheduled active fault-tolerant control of a wind turbine," *Journal of the Franklin Institute*, vol. 351, no. 7, pp. 3677 – 3706, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0016003213001798

[132] E. Kamal, A. Aitouche, R. Ghorbani, and M. Bayart, "Robust Fuzzy Fault-Tolerant Control of Wind Energy Conversion Systems Subject to Sensor Faults," *IEEE Transactions on Sustainable Energy*, vol. 3, no. 2, pp. 231–241, April 2012.

[133] A. A. Ozdemir, P. Seiler, and G. J. Balas, "Wind turbine fault detection using counter-based residual thresholding," in *18th International Federation of Automatic Control (IFAC) World Congress*, vol. 44, no. 1, 28 Aug.–2 Sep. 2011, pp. 8289 – 8294. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016449426

[134] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab, *Signals & Systems*, 2nd ed. Prentice-Hall Inc., 1997.

[135] Wikipedia. (2016) Support vector machine. [Accessed 16-May-2016]. [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine

[136] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A Practical Guide to Support Vector Classification," 2003.

Appendix A

Support Vector Machine (SVM)

The Support Vector Machine (SVM) is a relatively new supervised learning technique that is based on statistical learning theory that is introduced in 90's [123]. The SVMs are applied in classification and regression areas and have been widely used in data mining applications.

In the SVM, the basic idea is to map the data from the input space into a higher dimensional feature space, and then find an optimal hyperplane that maximizes the margin between the classes. SVMs are an example of two-class linear classifier, and the goal is to find a hyperplane that divides the two classes with the largest margin (Figure A.1). The decision function of the classifier is given by

$$f_{\mathbf{w},b} = \text{sgn}[\mathbf{w}^{\mathrm{T}}\mathbf{x} + b] \tag{A.1}$$

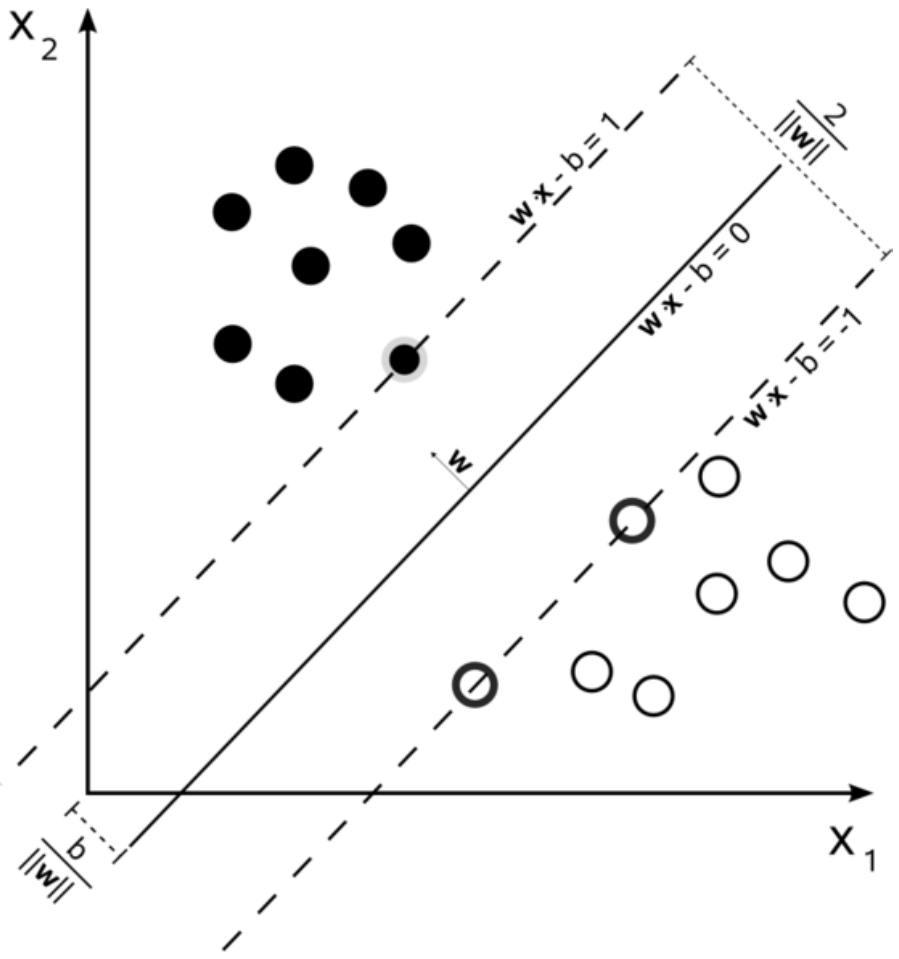


Figure A.1: Maximum margin hyperplane (reprinted from [135]).

In a training set where the data is linearly separable, and a hard margin (no slack allowed) is used, the support vectors are the points which lie along the supporting hyperplanes (the hyperplanes parallel to the dividing hyperplane at the edges of the margin). All of the support vectors lie exactly on the margin. Regardless of the number of dimensions or size of data set, the number of support vectors could be as little as 2.

In general, the the classification problem would be to find a hyperplane in a high

dimensional feature space Z, which divides the training instances (in the feature space) in a way that all the points within the same category on the same side of the hyperplane (Figure A.2). In this case, a map $\mathbf{z} = \Phi(\mathbf{x})$ is constructed from the input space $\mathbb{R}^n$ to a higher dimensional space Z and find the optimal hyperplane in Z such that the separation margin between positive and negative instances is maximized. As a result, the decision function of the classifier would be as follows

$$f_{\mathbf{w},b} = \text{sgn}[\mathbf{w}^{\mathrm{T}}\mathbf{z} + b], \tag{A.2}$$

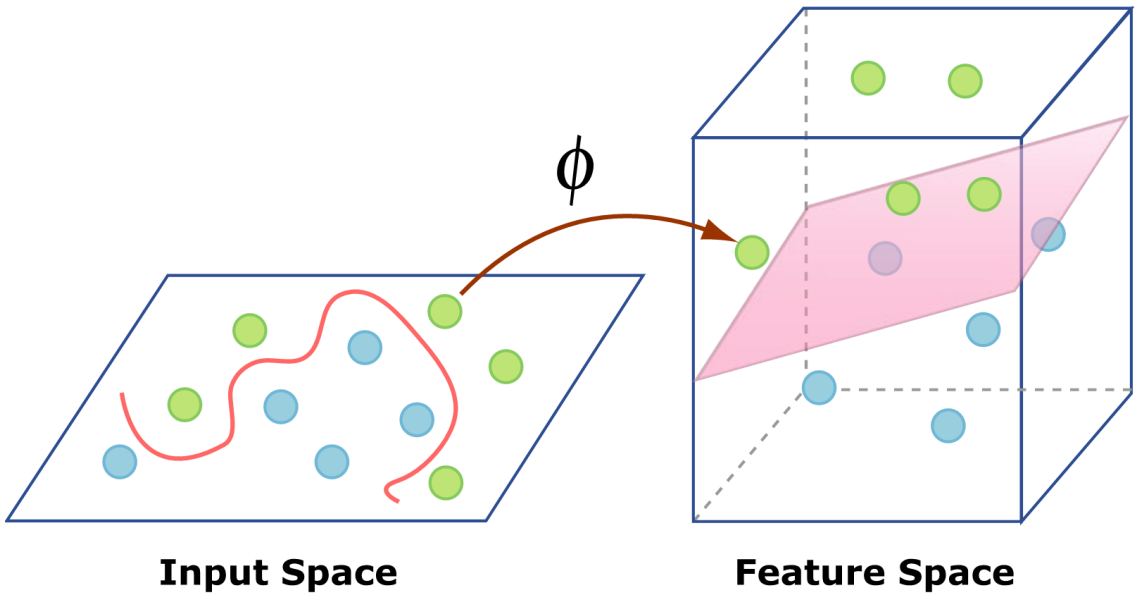


Figure A.2: Mapping data into higher dimensions (Image by MIT OpenCourseWare).

A.1 Formulation of SVMs

Given a training set of instance-label pairs $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, N$ where $\mathbf{x}_i \in \mathbb{R}^n$ (n is the dimension of the input space), and $\mathbf{y} \in \{-1, 1\}^N$, the SVM require the solution of the following optimization problem [136]:

$$\begin{aligned} \min \quad & \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^{\mathrm{T}}\Phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad i = 1, ..., N \\ & \xi_i \geq 0, \quad i = 1, \ldots, N \end{aligned} \tag{A.3}$$

where $C > 0$ is a regularization parameter for the trade-off between model complexity and training error, and the slack variable ξ_i measures the difference between $\mathbf{w}^{\mathrm{T}}\mathbf{z} + b$ and y_i. Due to the presence of the slack variable ξ_i, the above formulation is referred to as *soft margin* SVM.

Using the method of Lagrange multipliers for nonlinear constrained optimizations, we define the Lagrangian L as the objective function plus a linear combination of the constraints:

$$L = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i(1 - \xi_i - y_i(\mathbf{w}^{\mathrm{T}}\Phi(\mathbf{x}_i) + b)) - \sum_{i=1}^{N}\mu_i\xi_i \tag{A.4}$$

The Lagrangian dual problem for support vector learning is then given by

$$\begin{aligned} \max \quad & -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j (\Phi(\mathbf{x}_i)^{\mathrm{T}}\Phi(\mathbf{x}_j))\alpha_i\alpha_j + \sum_{i=1}^{N}\alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & 0 \le \alpha_i \le c, \quad i = 1,\ldots,N. \end{aligned} \tag{A.5}$$

The dual formulation of the SVM optimization problem depends on the data only through dot products. The dot product can therefore be replaced with a nonlinear kernel function, thereby performing large-margin separation in the feature space of the kernel. In other words, the kernel trick simplifies the quadratic optimizations used in support vector machines by replacing a dot product of feature vectors in the feature space with a kernel evaluation over the input space.

By introducing a kernel function $K(\mathbf{x},\mathbf{y})$ satisfying $K(\mathbf{x}_i,\mathbf{x}_j) = \Phi(\mathbf{x}_i)^{\mathrm{T}}\Phi(\mathbf{x}_j)$, the dual problem (A.5) becomes

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j K(\mathbf{x}_i,\mathbf{x}_j) - \sum_{i=1}^{N} a_i y_i \\ \text{subject to} \quad & \sum_{i=1}^{N} a_i = 0 \\ & -c_i^1 \le a_i \le c_i^2, \quad i = 1,\ldots,N \end{aligned} \tag{A.6}$$

where $a_i = \alpha_i y_i$, $c_i^1 = -c(\mathrm{sgn}(1-y_i))$, $c_i^2 = c(\mathrm{sgn}(1+y_i))$.

Note that the decision function given in (A.2) can be replaced by the following

function

$$f(\mathbf{x}) = \text{sgn}\left[\sum_{i=1}^{N} a_i \cdot K(\mathbf{x}_i, \mathbf{x}) + b\right]. \tag{A.7}$$

Some of popular kernels are given below

- Polynomial kernel $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^{\mathrm{T}}\mathbf{y} + 1)^p$,
- Gaussian kernel $K(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}$,
- Multilayer Perceptron kernel $K(\mathbf{x}, \mathbf{y}) = \tanh(\mathbf{x} \cdot \mathbf{y} + \sigma)$.

Consider a training set of pairs $(\mathbf{x}_i, y_i), i = 1, ..., l$ where $\mathbf{x}_i \in \mathbb{R}^n$ and the class labels $y_i \in \{-1, 1\}^l$. The SVM requires the solution to the following quadratic programming optimization problem [123]:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned} \tag{A.8}$$

where ϕ is the function that maps the training vectors $\mathbf{x}_i$ into a higher dimensional space. The regularization parameter $C > 0$ is a penalty factor applied to the error term. The direct solution to the optimization problem (A.8) is challenging due to the number of variables that are involved and the evaluation of the mapping function $\phi(\mathbf{x})$. Therefore, obtaining a solution to (A.8) is converted into solving a Lagrangian dual problem as

given by (A.9)

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j)) - \sum_{i=1}^{N}\alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{N}\alpha_i = 0, \\ & 0 \le \alpha_i \le c, \quad i = 1,\ldots,N, \end{aligned} \tag{A.9}$$

where a kernel function $K(\mathbf{x},\mathbf{y})$ satisfying $K(\mathbf{x}_i,\mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j)$ is introduced. The kernel function that is used here is the Radial Basis Function (RBF), and has the following description

$$K(\mathbf{x}_i,\mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i-\mathbf{x}_j\|^2}, \qquad \gamma > 0, \tag{A.10}$$

Note that this is the same as the Gaussian kernel, except the term γ is used instead of $\frac{1}{2\sigma^2}$ in the Gaussian kernel.

Appendix B

Supplementary Materials

Table B.1: Critical values for the two-tailed sign test for $\alpha = 0.05$ and $\alpha = 0.1$ levels of significance [126].

# Cases	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
$\alpha = \mathbf{0.05}$	5	6	7	7	8	9	9	10	10	11	12	12	13	13	14	15	15	16	17	18	18
$\alpha = \mathbf{0.1}$	5	6	6	7	7	8	9	9	10	10	11	12	12	13	13	14	14	15	16	16	17

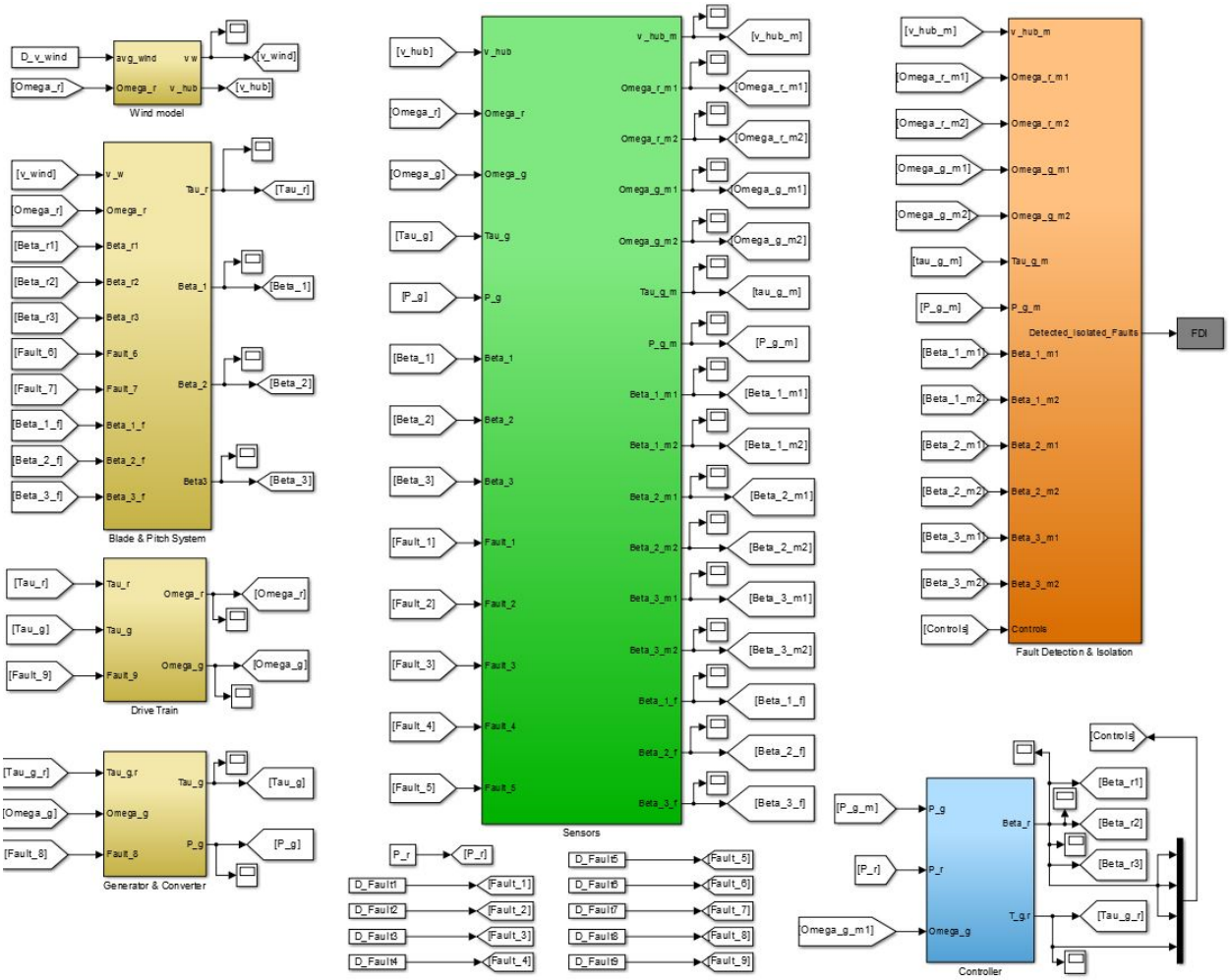


Figure B.1: A snapshot of the WT Simulink model.