IntechOpen

Fault Detection and Diagnosis

Edited by Constantin Volosencu

FAULT DETECTION AND DIAGNOSIS

Edited by **Constantin Volosencu**

Fault Detection and Diagnosis
http://dx.doi.org/10.5772/intechopen.76272
Edited by Constantin Volosencu

Contributors

Dusan Krokavec, Anna Filasova, Yu-Ling He, Yue-Xin Sun, Nageswara Rao, Bobby Philip, Marco Adonis, Atanda Raji, Ian Kuiler, Bilal Djamal Eddine Cherif, Azeddine Bendiabdellah, Mokhtar Bendjebbar, Souad Laribi

Notice

Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published chapters. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

First published in London, United Kingdom, 2018 by IntechOpen
eBook (PDF) Published by IntechOpen, 2019
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, The Shard, 25th floor, 32 London Bridge Street
London, SE19SG – United Kingdom
Printed in Croatia

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Fault Detection and Diagnosis
Edited by Constantin Volosencu

p. cm.

Print ISBN 978-1-78984-436-8

Online ISBN 978-1-78984-437-5

eBook (PDF) ISBN 978-1-83881-831-9

Meet the editor

Constantin Volosencu is a professor (full) at the "Politehnica" University from Timisoara, Department of Automation. He is the author of 10 books and 4 book chapters and also the editor of 5 books, author of over 150 scientific papers published in journals and conference proceedings, author of 27 patents and manager of several research grants. He is a member of several editorial boards of international journals, former plenary speaker, member in scientific committees and chair at international conferences. He has undertaken research in the field of control systems, electrical drives, power ultrasounds, fuzzy logic, neural networks, fault detection and diagnosis, sensor networks and distributed parameter systems. He developed electrical equipment for machine tools, spooling machines, high power ultrasound processes and other, with homologation of 18 prototypes and 12 zero manufacturing series.

Contents

Preface XI

Section 1 Hybrid Computing Systems 1

Chapter 1 **Fault Diagnosis of Hybrid Computing Systems Using Chaotic-Map Method 3**
Nageswara S. V. Rao and Bobby Philip

Section 2 Power Systems 31

Chapter 2 **Preventive Maintenance and Fault Detection for Wind Turbine Generators Using a Statistical Model 33**
Ian Kuiler, Marco Adonis and Atanda Raji

Chapter 3 **Hybrid Fault Diagnosis Method Based on Mechanical-Electrical Intersectional Characteristics for Generators 57**
Yu-Ling He and Yue-Xin Sun

Section 3 Power Electronics 75

Chapter 4 **A Comparative Study on Some Fault Diagnosis Techniques in Three-Phase Inverter Fed Induction Motors 77**
Bilal Djamal Eddine Cherif, Azeddine Bendiabdellah, Mokhtar Bendjebbar and Laribi Souad

Section 4 Kalman Filtering 99

Chapter 5 **Fault Residuals Based on Distributed Discrete-Time Linear Kalman Filtering 101**
Dušan Krokavec and Anna Filasová

Preface

This book offers a selection of papers in the field of fault detection and diagnosis, promoting new research results in the field, which come to join other publications in the literature. The book presents new coverage, starting from mathematical models for diagnosis for developing data and model-based strategies with analytical or statistical methods and new knowledge based-systems implemented on computer tests and diagnosis, applied in process control with applications. Authors from countries of four continents: United States of America, South Africa, China, India, Algeria and Croatia published worked examples and case studies resulting from their research in the field of fault detection and diagnosis. The readers are provided with new solutions and answers to questions related to the emerging fault detection and diagnosis principles and their implementation. This book will be of interest and useful to a large number of persons: graduate students of engineering, researchers in the field of fault detection and diagnosis, faculty staff and managers who want to understand fault detection issues and their economic benefits. With an open access publication, the book may have visibility, worldwide researchers may read, download and interact with the published content. The book offers relevant information to further development of new projects. Fault detection and diagnosis has a great importance in all industrial processes, to assure the monitoring, maintenance and repair of the complex processes, including all hardware, firmware and software. Based on these considerations, the book may have a large impact in the scientific community.

In a brief description, the book has four sections, determined by the application domain and the methods used: 1. Hybrid Computing Systems, 2. Power Systems, 3. Power Electronics and 4. Kalman Filtering. In the first section, the readers will find a technical report on fault diagnosis of hybrid computing systems, based on the chaotic-map method that uses the exponential divergence and wide Fourier properties of the trajectories, combined with memory allocations and assignments. In the second section, two chapters are included: one of them presents a study on preventive maintenance and fault detection for wind turbine generators using statistical models and the second chapter presents a technical report on fault diagnosis for turbo-generators, based on the mechanical-electrical intersectional characteristics. The third section contains a technical report that presents some techniques of detection and localization of open-circuit faults in a three-phase voltage source inverter fed induction motor. The fourth section presents a theoretical study on the application of distributed discrete-time linear Kalman filtering with decentralized structure of sensors in fault residual generation. The published applications presented by the authors are indicative of their interest and engagement. The book shows the degree of information of authors in the field and their dedication, appreciation and enthusiasm of the field they care about. The authors choose to publish their research project results to explain fault detection and diagnosis methods in different applications, to explain variations of faults in different industrial processes, to test the consequences of faults on processes, they conducted experimental tests of hypothesis, developed mathematical versions of process models and use fundamental ideas to address applied problems in novel ways.

The editor wishes to thank the research authors of the chapters for their scientific contribution. The chapters were edited and published following a rigorous selection process, out of more than triple the number of publication proposals. Also, it is a pleasure to thank and acknowledging the help of many individuals from the editorial process that have made this book possible. The publishing provided an efficient set of editorial standards, which ensured the quality of the scientific level of relevance of the accepted chapters.

Prof. Constantin Volosencu
"Politehnica" University from Timisoara
Romania

Section 1

Hybrid Computing Systems

Chapter 1

Fault Diagnosis of Hybrid Computing Systems Using Chaotic-Map Method

Nageswara S. V. Rao and Bobby Philip

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.79978

Abstract

Computing systems are becoming increasingly complex with nodes consisting of a combination of multi-core central processing units (CPUs), many integrated core (MIC) and graphics processing unit (GPU) accelerators. These computing units and their interconnections are subject to different classes of hardware and software faults, which should be detected to support mitigation measures. We present the chaotic-map method that uses the exponential divergence and wide Fourier properties of the trajectories, combined with memory allocations and assignments to diagnose component-level faults in these hybrid computing systems. We propose lightweight codes that utilize highly parallel chaotic-map computations tailored to isolate faults in arithmetic units, memory elements and interconnects. The diagnosis module on a node utilizes pthreads to place chaotic-map threads on CPU and MIC cores, and CUDA C and OpenCL kernels on GPU blocks. We present experimental diagnosis results on five multi-core CPUs; one MIC; and, seven GPUs with typical diagnosis run-times under a minute.

Keywords: fault diagnosis, hybrid systems, chaotic maps, multi-core CPU, GPU

1. Introduction

High performance computing systems utilize increasingly complex *hybrid* nodes that consist of multi-core central processing units (CPU) combined with many integrated core (MIC) or graphics processing unit (GPU) accelerators [1]. The next generation systems that target Exascale computations are expected to be massive with computing elements totaling a million [2, 3]. Furthermore, these computing systems are expected to be built, at least in part, using off-the-shelf components such as CPUs, Accelerated Processing Units (APU) and GPUs, which have an expected life-span in the range of 5–10 years. Consequently, the computations that run for a few hours on such systems are likely to experience multiple faults, and it is essential to account for them to achieve the *resilience* of these computations [4, 5]. Detection of such faults contributes to resilient computations in a number of ways such as: supporting the quarantine of faulty units from the scheduler pool; replacement of faulty processor boards and accelerators; and, initiation of application migration and check point recovery. However, fast and efficient detection of such faults in hybrid computing systems is complicated due to the continued increases in the number and complexity of processors, accelerators and interconnects.

However, fast and efficient detection of such faults in hybrid computing systems is complicated due to the continued increases in the number and complexity of processors, accelerators and interconnects.

The impact of such faults could be quite significant on certain scientific computations, particularly if they fail to trigger checkpoint recovery or process migration, or result in too many errors that require inordinate number of checkpointing operations. Furthermore, the variety of faults is expected to expand in future as hybrid architectures evolve with increasing numbers of cores, sockets and blocks, and with complex processors and interconnect designs. The component faults in these systems can manifest in a variety of ways: faults in arithmetic and logic units (ALU) and floating point units (FPU) lead to instruction execution errors; and memory element errors and transport errors (over memory bus, inter-processor link, PCI bus connection to GPU memory, or interconnect) lead to erroneous data. In production systems, current support for detecting these faults is somewhat limited, primarily to hardware monitors and codes with "known" outputs, and several other approaches are currently under development [5–18]. In fact, some faults that develop during the computation may not be detected at all, and the computation may indeed run to completion and produce unsuspected erroneous output. One practical diagnosis technique is to run an application and compare its output with *a priori* known correct values. For example, codes such as CUDA-enhanced HPL [19] with known outputs have been used in practice to verify error-free executions. These codes, however, require significant execution times, since they solve dense linear systems with a primary purpose of benchmarking the (error-free) system.

We propose lightweight codes to quickly and efficiently detect component faults in hybrid computing nodes consisting of multi-core CPUs with MIC or GPU accelerators. These codes are based on developing the chaotic-map method[1] to diagnose hybrid systems, and are among the smallest codes capable of detecting a large class of ALU, memory and interconnect errors, typically requiring a few iterations of few instructions. The chaotic-map method is introduced [15] as a fault diagnosis tool for computing systems, and applied to multi-core CPUs using pthreads in [16]; but these codes are not transferable to GPUs due to their significantly different architectures and software support. We first extend the chaotic-map implementation to include logical and integer operations, and develop CUDA and OpenCL kernels to diagnose

[1] Chaotic maps have origins in the analysis of non-linear systems with complex dynamics, such as weather systems. Extensive theory and analysis methods of chaotic maps have been developed [20], and are often used for establishing the existence of chaotic dynamics in a wide range of non-linear system models [21].

Multi-Core CPU:

4-core Intel Xeon 2.67 GHz

16-core AMD Opteron 2.3 GHz

16-core Intel Xeon 2630

32-core Intel Xeon E5-2650 2.7GHz

48-core AMD Opteron 6176 SE 2.29GHz

Single-MIC:

Intel Xeon Phi Coprocessor 3120P/A

Single-GPU:

Quadro 600, Quadro K4200, Tesla T10, Tesla C1060

Tesla K20X, Tesla K20c, AMD Firepro W9000

Single-APU:

AMD A10-7850 K

Multiple-GPU:

8 Nvidia Tesla T10 GPUs

Nvidia Tesla K20c and AMD Firepro W9000

Intel HD Graphics 4000 and Nvidia GeForce GT 650 M

Table 1. Nodes used in implementation and testing of diagnosis codes.

GPUs, and integrate them with pthreads multi-core CPU diagnosis codes to diagnose large systems with hybrid nodes. We have implemented and tested these diagnosis codes on systems shown in **Table 1**, namely on five multi-core CPUs; MIC accelerator; seven GPUs; and, three multi-GPU systems.

Our main objective is to rapidly diagnose the faults in large-scale hybrid computing systems with the architecture shown in **Figure 1**. In particular, we consider detecting component faults entirely by software means, similar in spirit to the approaches of Erez et al. [6] and Sahoo et al. [18]; in particular, we focus on codes that run in a few minutes to diagnose non-transient faults.

A finer diagnosis to pinpoint individual faulty digital gates, as typical in the fault diagnosis literature [22], requires solutions to the underlying NP-hard problems. The general problem of detecting resilience of codes is computationally undecidable in Turing sense [17]. Also, sporadic faults that last for short durations (i.e., micro seconds) are not addressed here. Our diagnosis codes are intended to provide "quick" diagnosis to complement other methods[2] such as hardware monitors, HPL codes [19], application-specific detection methods [23–26], and verification systems [27]. While our original motivation is to support facility operations, our diagnosis codes can be made part of a broader, resilience ecosystem to complement and

[2] Due to the multi-disciplinary nature of the area of extreme-scale resilient computations, the literature on related works is quite extensive, and we only refer to a very small set of works that are directly connected to the technical areas of this report.

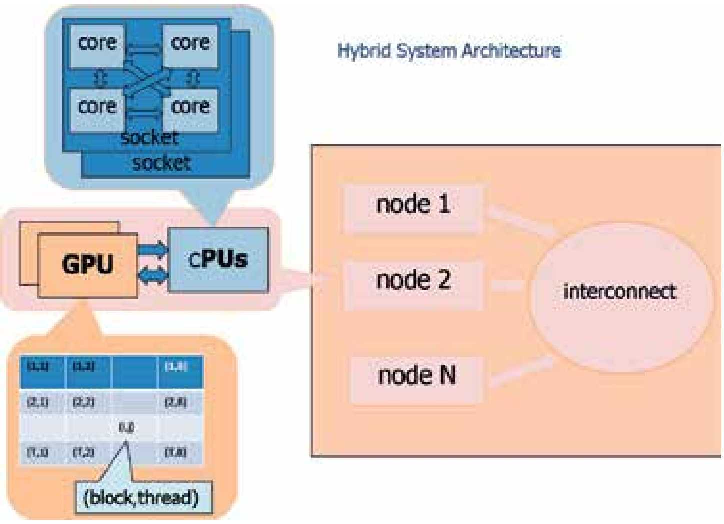


Figure 1. CPU-GPU hybrid system architecture.

support Algorithm-Based Fault Tolerance (ABFT) methods [7, 28]; software-based fault detection [5, 6]; and, likely invariants for detecting hardware faults [18].

Our overall approach is to compute chaotic-map trajectories concurrently on all CPU and MIC cores, and GPU blocks. Two properties of chaotic map trajectories are critical for diagnosis purposes: (a) their *exponential divergence* ensures that the trajectories subjected to faults will rapidly diverge from the majority (fault-free) and are easily detected; and (b) their *density* and *aperiodicity* ensures that they span across and cover a majority of bits involved in the constituent operations. Our codes utilize concurrent threads to compute chaotic map trajectories that are augmented with: (a) arithmetic and logic operations to diagnose ALU operations, and (b) content-preserving data movement operations to diagnose memory elements, busses and interconnects.

This paper is organized as follows. We describe the basics of the fault detection method using chaotic maps in Section 2. We present a brief description of the hybrid architecture and the details of our diagnosis codes in Section 3. We describe the overall diagnosis method in Section 3.1, and provide the details of diagnosis of CPU and MIC cores in Section 3.2, and the details of diagnosis of GPU blocks using CUDA and OpenCL kernels in Sections 3.4 and 3.4, respectively. We present experimental results in Section 4.

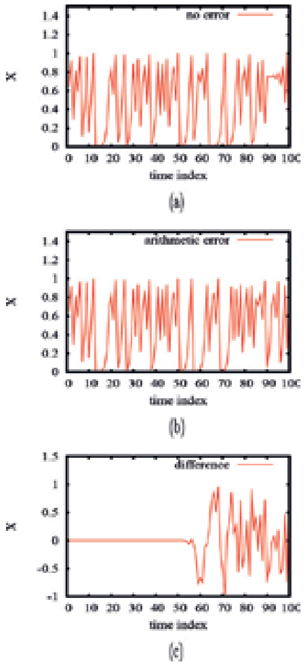
2. Diagnosis using chaotic maps

A *Poincare map* $M : \mathfrak{R}^d \mapsto \mathfrak{R}^d$ specifies a sequence, called the trajectory, of a real-vector *state* $X_i \in \mathfrak{R}^d$ that is updated at each iteration i such that $X_{i+1} = M(X_i)$ [21]. The computation of $M(X_i)$ may involve floating-point operations, such as multiplication and addition, and logical operations such as comparison of numbers, and can vary significantly in the number and types of operations. The trajectory $X_0, X_1, \ldots, X_t$, such that $X_i = M^i(X_0)$, generated by certain Poincare maps can exhibit complex profiles, even when M is computationally simple. In **Figure 2**, we show trajectories of the *logistic map* $M_{L_a}(X) = aX(1 - X)$, for $X \in [0, 1]$, which requires two multiplications and one subtraction per iteration. In **Figure 3**, we show the trajectories of the *tent map*

$$M_{T_b}(X) = \begin{cases} bX & \text{if } X \leq 1/2 \\ b(1 - X) & \text{if } X > 1/2 \end{cases}$$

for $X \in [0, 1]$, which requires a comparison operation, one multiplication and at most one subtraction per iteration. The trajectories of these maps exhibit complex dynamics as shown in **Figures 2(a)** and **3(a)** for the logistic map for $a = 4$ and the tent map for $b = 2$, respectively. The trajectories generated by the Poincare map M are characterized by the *Lyapunov exponent* defined as $\mathcal{L}_M = \ln\left|\frac{dM}{dX}\right|$, which characterizes the separation of the trajectories that originate from the nearby states. For example, the Lyapunov exponent of the tent map is $\mathcal{L}_{M_{T_b}} = \ln b$, which is defined for all $X \in [0, 1]$ except at $X = 1/2$.

A bounded trajectory $X_0, X_1, \ldots$ generated by the Poincare map $M(.)$ is *chaotic* if (i) it is not asymptotically periodic, and (ii) Lyapunov exponent $\mathcal{L}_M$ is greater than zero [21]. Two important properties of the chaotic maps are exploited here for fault diagnosis: (a) the *exponential divergence* ensures that trajectories whose states slightly differ from each other at any iteration



(a)

(b)

(c)

Figure 2. Trajectories of the logistic map. (a) trajectory with no errors, (b) trajectory under arithmetic error, (c) differences in trajectories with and without error.

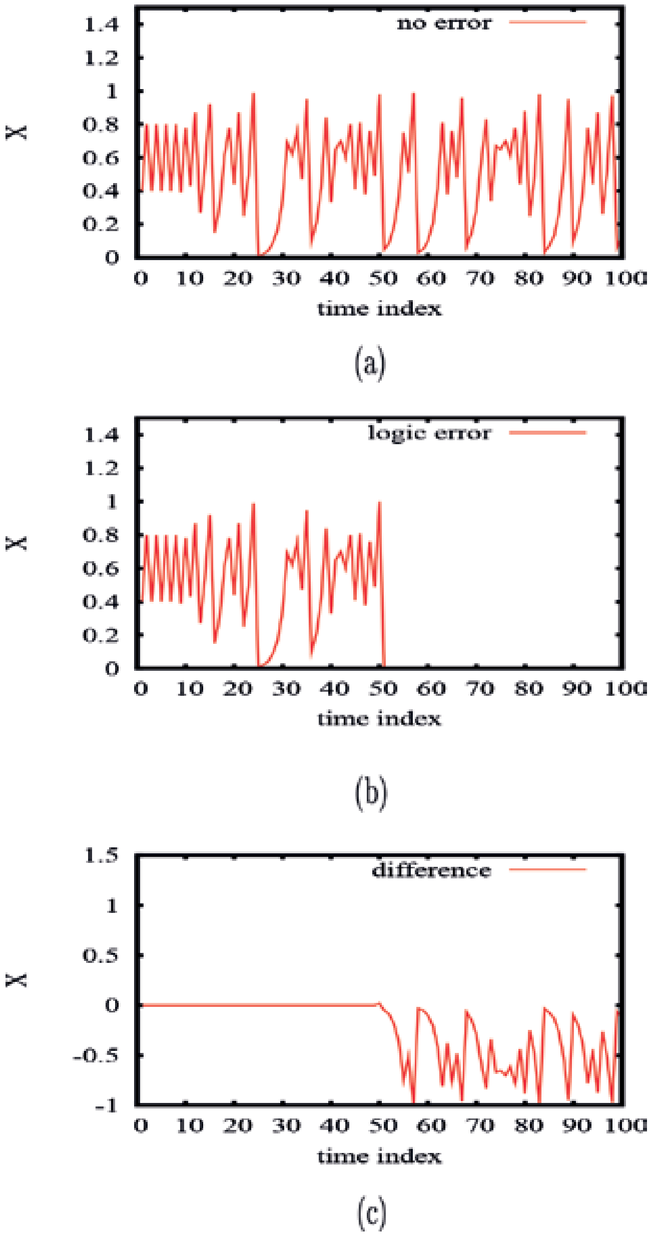


Figure 3. Trajectories of the tent map. (a) trajectory with no error, (b) trajectory with logic error, (c) difference in trajectories with and without error.

rapidly diverge within a few steps, and (b) the *high spatial density* and *broad Fourier spectrum* of states of a trajectory spreads them across the bit-space of the underlying computing operations within a few iterations. The first property has been proposed in [15] as an effective computational mechanism to rapidly amplify errors caused by factors such as bit flip in memory content or stuck-at fault in an ALU operation. We extend this approach to diagnose High

Performance Computing (HPC) systems with small fault rates: these maps are computed concurrently so that errors are detected by comparing them to a small majority of them. In addition, several chaotic maps have very small computational requirements, and a vast literature is available on the analytical [8, 20], statistical [29] and computational aspects of these maps [13, 30]. It is possible in theory to utilize linear maps in a similar way, but they are not as efficient in detecting "small" errors (such as in least significant digits) which have to be linearly amplified through multiple iterations to trigger detection, and also they do not generate dense states and hence are limited in their bit-level coverage.

The trajectories that slightly differ from each other in any iteration rapidly diverge from each other in a few steps, as shown in **Figure 2(a)** and **(b)** for the logistic map, and **Figure 3(a)** and **(b)** for the tent map. This property is utilized as a mechanism to rapidly amplify errors in computations caused by factors such as bit flip in memory elements or stuck-at fault in an ALU operation. Also, through the iterations, the states are spread across the interval $[0, 1]$ so that the bit-space of the underlying computations, for example, of the registers, is covered with a high likelihood. The difference between two trajectories with the same starting state is shown in **Figure 2(c)** for the logistic map, where the state is corrupted by 1/10000 magnitude in iteration 50. During iterations 0 through 50, the difference between the trajectories is 0, but the small difference in state magnitude is amplified to above 0.25 within 8 iterations, which is typically under 1 ms on the systems we tested. In **Figure 3**, we show the effect of error in the logical operation, wherein the result of the comparison is flipped in iteration 50. The effect on the trajectory is more dramatic as shown in **Figure 3(b)**, and the difference in the trajectories crosses 0.25 within two iterations. Such divergence in trajectories can be detected by a magnitude test, and the detection time is controlled by the Lyapunov exponent of the map. While both logistic and tent maps provide exponential divergence, they cover the state space [0, 1] differently as illustrated in **Figure 4**, as a result of different Lyapunov exponents. It is lower in the middle and large at the ends of state space [0, 1] for the logistic map, but is uniform for the tent map, which makes it preferable for its coverage to support diagnosis.

The computation of a chaotic map $M(.)$ is sensitive to errors in its constituent operations, and the mechanisms used in storing and updating the states. The detectable faults include errors in arithmetic and logical operations performed by ALU, and faults in registers and memory, but

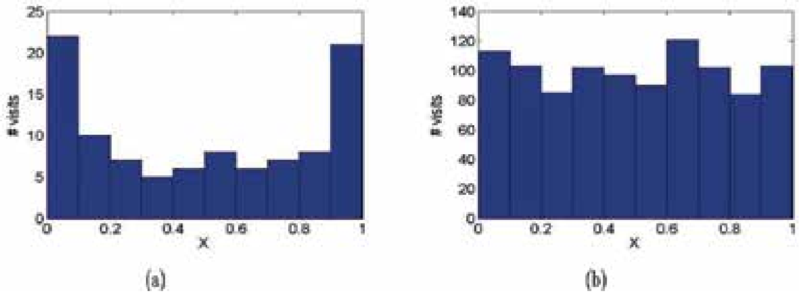


Figure 4. State space coverage by chaotic trajectories. (a) coverage under logistic map, (b) coverage under tent map.

are limited to the basic operations of $M(.)$ itself. We propose the *chaotic-detection map* $M_D(.)$[3] that augments the detection space of $M(.)$ so that its execution path is routed through different computing operations, memory locations and interconnect links to capture errors due to them. The chaotic-detection map is obtained by adding the following two types of operations to the chaotic map iterations.

a. **Augmenting computing operations:** The chaotic map iterations are augmented with sequences of logical and arithmetic operations, which are selected based on the instruction sets of CPU cores and GPU blocks to complement the original chaotic map operations. The same sequence of operations is used in all concurrently computed maps by using the same process code for pthreads of all CPU cores and the same CUDA kernel on all GPU blocks. The type of state variable is used to exercise different parts of the computing units; in particular, it is scaled to a large integer and addition and multiplication are applied to exercise integer operations. Also, special operations such as log are applied to X_i to exercise special instructions that are implemented in hardware by Extended Math Unit (EMU), when applicable. The type casting of variables is used to exercise single precision and double precision processing units as well as Vector Processing Units (VPU), as described in the next section.

b. **Content-preserving data movement operations:** The state variable X_i is moved among the memory elements and/or across the interconnects in between applying $M(.)$ iterations, to capture errors in the memory elements and paths, and during the transmission across the interconnects. In each operation, the contents of X_i are unchanged under failure-free conditions. These movements can be realized by several means based on the supported operations, ranging from simple assignment statements to employing additional variables in the "shared" memory to utilizing explicit MPI, CUDA or other constructs. In particular for multi-core processors, memory assignments using pthreads can be used for both purposes, namely, to test memory unit errors as well as transport errors across the memory bus or hypertransport.

The rate of divergence of a chaotic map, and hence the detection times of failures depend on the Lyapunov exponent $\mathcal{L}_M$, generally larger values leading to quicker divergence. The class of faults detected by a chaotic-detection map depends on the chaotic map and the augmenting and data movement operations as well as the computing units used for their computation. A main consideration in developing the diagnosis codes is to efficiently compute the chaotic-detection maps on computing units with identical parameters, sequences of augmenting operations, state-preserving movement operations and chaotic map updates, so that the end states are identical under fault-free conditions. Their implementation critically depends on the software primitives supported on the systems, and they in turn are closely tied to the underlying system architecture, including the location of the computing elements, memory hierarchies and interconnects. In the next section, we describe specific implementations to compute chaotic-detection maps on multi-core CPUs, MICs, GPUs and hybrid systems.

[3] The chaotic-detection map is a generalization of the chaotic-identity map proposed in [15], which was restricted to the operations with inverses.

3. Hybrid system diagnosis

We consider hybrid computing systems, wherein each node consists of muti-core, possibly multi-sockets CPUs, and one or more GPU and XeonPhi MIC accelerators; in the limiting case, we have a single node with a multi-core CPU and zero or more GPUs. Computations are spawned to run on CPU cores using OpenMP, pthreads or similar constructs, and on GPU blocks using CUDA or OpenCL threads. Within each node, however, the data movements are carried out differently on CPUs and GPUs, since the former accesses different levels of on-board memory, but the latter can only directly access memory physically located on the GPU. The CPU-GPU data transfers are realized using CUDA or OpenCL in our case using memory copy operations between CPU on-board memory and GPU device memory. Computations on GPUs utilize thread bundles on GPU blocks using kernels written in CUDA C or OpenCL. Kernels are launched from the CPU of a node onto the corresponding GPU blocks as a collection of threads. In this section, we describe different components of the diagnosis codes for hybrid systems based on the chaotic-detection maps. Since these can be used as stand-alone codes for simpler systems, codes for a single-node with multi-core CPU with zero or more GPU accelerators, or as a cluster with only CPUs, are presented in separate sections.

3.1. Node diagnosis module

The overall diagnosis strategy is to utilize the "reliable" nodes to launch a *node diagnosis module* on each node as shown in **Figure 5**, under the working assumption that only a small number of

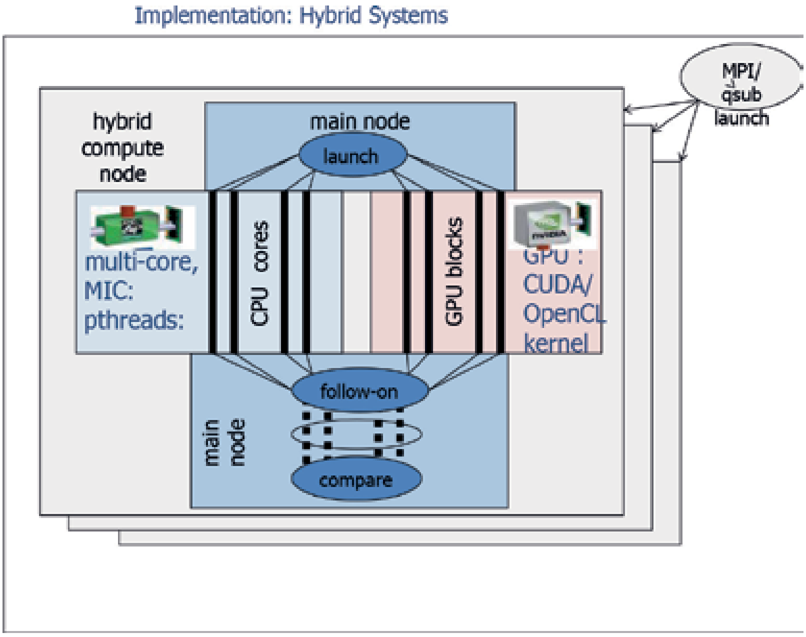


Figure 5. Diagnosis approach for hybrid computing systems.

nodes contain faulty components and a majority of them are fault-free. The node diagnosis module is written in C, and consists of a CUDA or OpenCL C kernel for GPUs, and pthreads code for multi-core CPUs or MIC accelerators. On each node, the diagnosis module detects the number of cores (more precisely, the processing units) using linux/proc system, and also explicitly checks for the physical presence of GPUs using CUDA C or OpenCL system calls (to avoid GPU emulations). It then allocates and initializes node-level global memory and copies the contents onto on-board device memory of GPUs connected to the node. Next, from the default CPU core, it launches concurrent threads to compute the chaotic-detection maps on the computing units, namely CPU cores and GPU blocks, and gathers their outputs and generates the diagnosis output.

The chaotic-detection map computation carried out by threads on each CPU core and GPU block consists of the following basic steps, which will be customized to CPU and GPU architectures (as described in the next sections):

i. Local memory is allocated and initialized based on the specifics of CPU or GPU;

ii. Initial state X_0 of the chaotic-detection map, and additive and multiplicative factors, denoted by A and M respectively, the numbers of pre and post iterations N_1 and N_2, respectively, are accessed so that all threads use the same values;

iii. N_1 iterations of the chaotic-detection map are computed using starting state X_0 to obtain X_K. followed by a sequence of augmenting operations, for example, addition and subtraction of the additive factor A, and division and multiplication with factor M, namely,

$X_K = X_K + A;$

$X_K = X_K - A;$

$X_K = X_K / M;$

$X_K = X_K * M,$

to check addition, subtraction, multiplication and division operations.

iv. A fixed sequence of content-preserving data movement operations are performed on variable X_K that are specific to CPU or GPU; and

v. N_2 iterations of the chaotic-detection map are computed with starting state X_K to obtain final state X_E.

At the completion of threads, outputs X_E's from all concurrent threads are transferred back to the default CPU core and are used as starting states for N_3 iterations of a follow-on chaotic map. This follow-on chaotic map amplifies the errors captured by the outputs of chaotic-detection maps from the CPU and GPU threads as well as those occur during data transfers, for example, from GPU to CPU over the PCI bus.

The final outputs X_F's of the follow-on chaotic map are compared to a pre-computed correct state (or to the majority of a small subset of them). If X_F's of all the threads match then no error is declared. If not, diagnosis steps (iii)–(iv) are executed separately to identify portions during which errors occurred. The step (iii) identifies ALU errors in executing +, −, / and * operations, and other operations of interest can be added. Steps (i), (ii) and (iv) are customized to match the memory architectures of CPU and GPU as described in the following sections, wherein assignment operations are used to diagnose memory elements and data paths. The memory and interconnect diagnosis codes described here, however, are limited mainly to illustrate the detection of faults in memory elements and data paths rather than complete sweeps of memory and interconnects.

All threads that compute chaotic-detection maps are provided identical parameters in step (ii). These parameters are setup on global arrays $GM[.,.]$ of size S_G on each node, which are accessible to all CPU cores and are explicitly transferred to GPU memory. For initialization on the node, malloc call is used for allocating the memory and memset is used to fill the memory with the values. The data movements in step (iv) do not alter the contents if there are no errors in storage or transfer operations, but are designed specifically to match the memory architecture of CPU and GPU; in particular, primitives such as assignments can be used to diagnose memory and transfer errors as will be described in next sections.

3.2. Multi-core CPU diagnosis

Multi-core CPU systems are composed of one or more sockets, each housing a number of processor cores connected to memory modules, which are typically organized in a hierarchy. An example of a single socket quad-core system is HP Z400 workstation shown in **Figure 6** consisting of four 2.67 GHz Intel Xeon CPUs. The cache memory units are connected over the memory bus such that L1 caches are local to processor cores, L2 caches are shared between pairs and L3 caches are global. The memory caches are connected over a combination of memory bus and hypertransport links. The L1 cache is local to cores whereas global memory is accessible to certain cores via hypertransport links. Thus, certain memory transfers between local and global memory take place over hypertransport links.

We now provide the details of node diagnosis module that is specific to multi-core CPUs. It partitions the global memory into non-overlapping parts assigned to N_C processor cores, and launches dedicated threads one on each core as shown in **Figure 7**. Processor core i is assigned the subarray $GM[i,.]$ of size S_{G_i}. Then, a single thread is invoked on each core i using pthread_setaffinity_np call, and this thread computes the chaotic-detection map with the following expanded steps described in Section 3.1.

(i) Local memory is allocated and initialized within the thread as an array $LS[.]$ of specified size S_L using malloc and memset.

(iv-a) The variable X_K is stored and retrieved from each element of the initialized local memory:

```
for $j = 1, \ldots, S_L$.
    $LS[j] = X_K$;
    $X_K = LS[j]$;
```

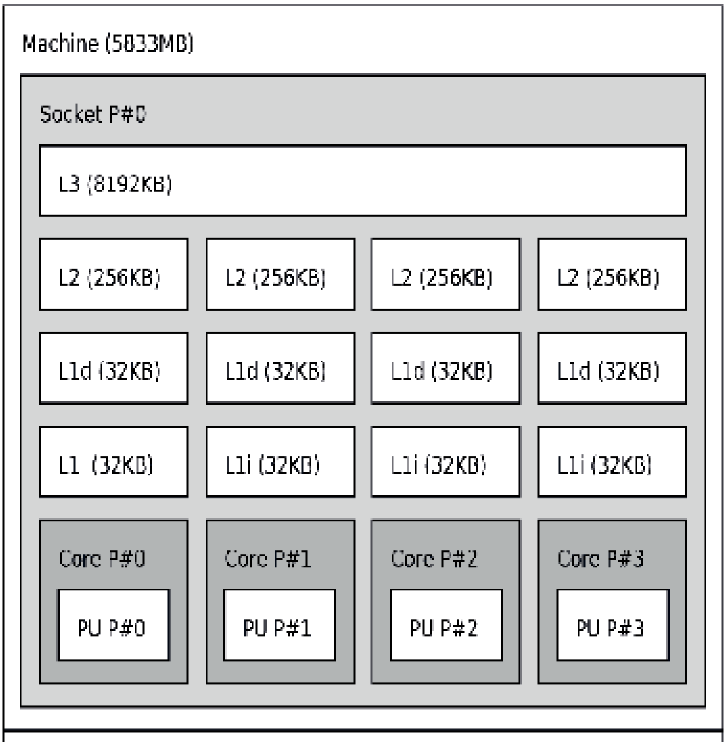


Figure 6. Architecture of 4-core HP Z400 workstation.

These operations are carried out between core *i* and its local memory.

(iv-b) The variable X_K is stored and retrieved from each element of the assigned partition of the global memory such that the thread assigned to processor code *i* executes the following code.

```
for $j = 1, \ldots, S_{G_i}$.
        $GM[i, j] = X_K;$
        $X_K = GM[i, j];$
```

The basic idea of steps (iv-a) and (iv-b) is to utilize the variable assignments to diagnose both memory elements as well as data paths. The step (iv-a) exercises the local memory operations and detects errors in the memory elements, as well as during transport by the memory controller. The step (iv-b) exercises the processor interconnect, as well as the elements in global memory; the interconnect is memory bus for HP z400 workstation, and hypertransport for HP Proliant server. While these steps do not cover all possible errors, they are likely to capture several major errors in ALU, memory and interconnect so that processors with detected errors can be excluded from computations or their boards may be replaced. This memory diagnosis part can be further refined: (a) NUMA tools can be utilized to explicitly allocate memory in different locations and layers so that the assignment operations require data to be transferred across the memory connections, and (b) assignment primitives under higher level constructs

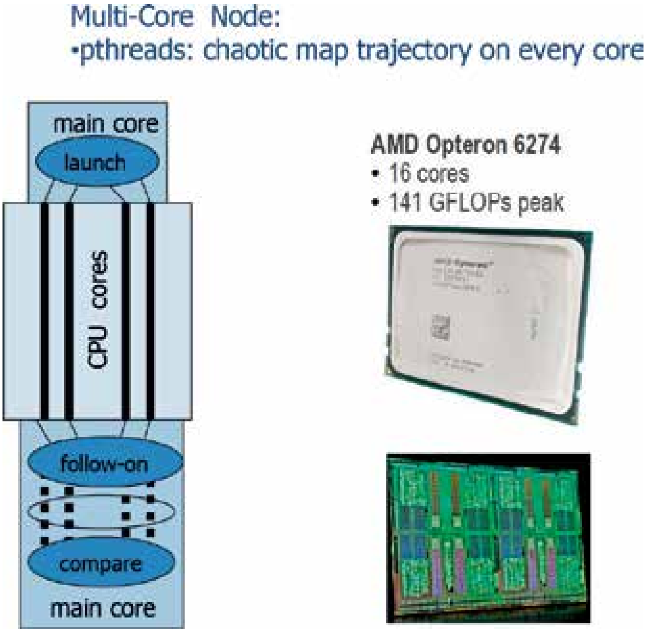


Figure 7. Diagnosis of CPU cores.

such as openMP, SHMEM or PGAS can be used as diagnose memory paths at much higher level using fairly simple codes.

3.3. Xeon phi diagnosis

Intel's first generation MIC Architecture, code named Knights Corner has 57–61 cores per coprocessor. The particular coprocessor we had access to was the Intel Xeon Phi 3120P/A coprocessor with 57 cores, 1TF double precision performance, 6GB GDDR memory with 240 GB/s data transfer rates and a 1.1 GHz clock. Internally the 3120P consists of 57 cores with each core having a VPU, a x87 math co-processor providing double precision transcendentals (non-vectorized) and a scalar processing unit. The VPU on each core internally consisted of 8 double precision FPUs and 16 single precision (SP) FPUs and an extended math unit (EMU) providing single precision vectorized transcendental functions. The VPUs are capable of 8 DP or 16 SP operations per clock cycle. Associated with each VPU are four hardware thread execution contexts each having access to 32,512 bit wide private registers (zmm0-zmm31)and 8 16 bit mask registers, 7 of which are writeable. Associated with each core are 32 KB L1 data and instruction caches and a unified 512 KB instruction and data L2 cache. The instruction set associated with the VPUs is the Intel Initial Many Core Instruction (IMCI) set. The 57 cores are on a round robin bidirectional ring interconnect with 8 memory controllers. For further details we refer to [14, 31–33].

The Xeon Phi node diagnosis module tests the single and double precision ALUs within each VPU, the x87 math coprocessor for each core, the EMU for each core, the general purpose vector registers for each hardware thread, and the L1 and L2 caches. We did not attempt to design explicit tests for the memory controllers, RAM, or the interconnects to the CPU within this module though in theory this should also be possible. The code is written in C and IMCI assembly. The icc compiler was used to automatically generate vectorized code for the VPUs. The code can be executed either by offloading to the coprocessor or by natively executing on the coprocessor. pthreads is used to spawn off one thread for each logical core detected. For the Intel Xeon Phi 3120P/A coprocessor that we had access to this worked out to a total of 57 × 4 threads corresponding to the 4 hardware threads associated with the 57 cores. The core affinity for each thread was explicitly set using the **pthread_attr_setaffinity_np** call. Each thread executes a set of chaotic-map detection routines based on the description in Section 3.1 customized to test the hardware components listed above. In particular, it was necessary to write both single and double precision chaotic map routines that operated on 64 bit aligned arrays in order to exercise the SP and DP ALUs. Testing of the ×87 math co-processor was done through the compiler switches **-mmic -fp-model strict** that disabled auto-vectorization and forced the ×87 coprocessor to be exercised. This was verified through examining the generated assembly code. Diagnosis of the EMU was ensured by introducing transcendental function calls. From examining the generated IMCI assembly it was clear that the generated code did not exercise all 32 of 512-bit vector registers (zmm0–zmm31) and mask registers associated with each thread. In order to test the registers, assembly routines were written to span all registers associated with each thread that performed the chaotic map iterations.

3.4. GPU diagnosis using CUDA kernels

Nvidia general purpose GPUs (GPGPU) can be viewed as a set of streaming multiprocessors (SMs) [34] as shown in **Figure 8**. Each SM internally consists of a number of simplified cores: CUDA cores which consist of scalar SP floating point and arithmetic ALUs, DP cores, Special Function Units (SFU) for transcendental functions, and Load/Store units. The number of cores and the relative ratio of the different types has varied from generation to generation. Each SM has a number of schedulers and instruction dispatch units associated with it, as well as a register file shared by all cores in the SM, and local memory partitioned as shared memory and L1 cache. SMs also have access to global device memory. The basic scheduling unit for Nvidia GPGPUs is a warp which consists of 32 threads which operate in SIMT (single instruction multiple thread) fashion. At a higher level, threads are organized into thread blocks and on each block all threads execute concurrently as shown in **Figure 9**. These computations have only direct access to memory on the device DRAM with support from caches. **Table 2** lists hardware specifications for some of the Nvidia microarchitecture generations [9–12]. We note that we have tested on a variety of Nvidia GPGPUs including Quadro 600, Tesla C1060, Quadro K5000, and Tesla K20X (**Table 2**).

The diagnosis module for Nvidia GPUs performs separate chaotic-map detection iterations to detect faults on the CUDA cores, the DP cores, and the SFU components. The global memory $GM[.,.]$ is copied by the node diagnosis module onto the device memory $GM_G[.,.]$ to make it accessible to GPU threads. The thread computations are implemented by a CUDA kernel that is loaded and executed on GPU(s). The same kernel code is executed on each block, which

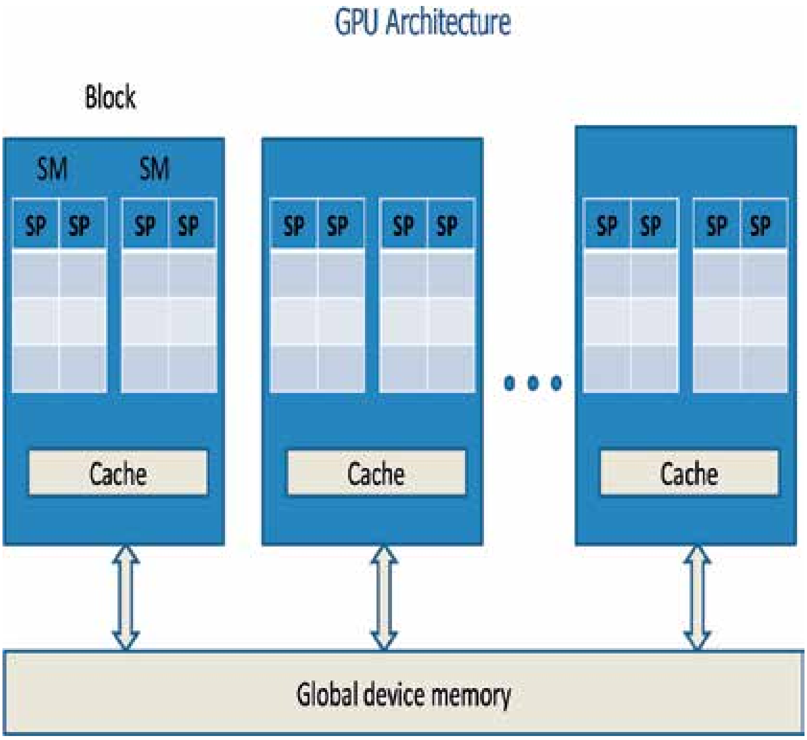


Figure 8. Architecture of a CUDA-capable GPU.

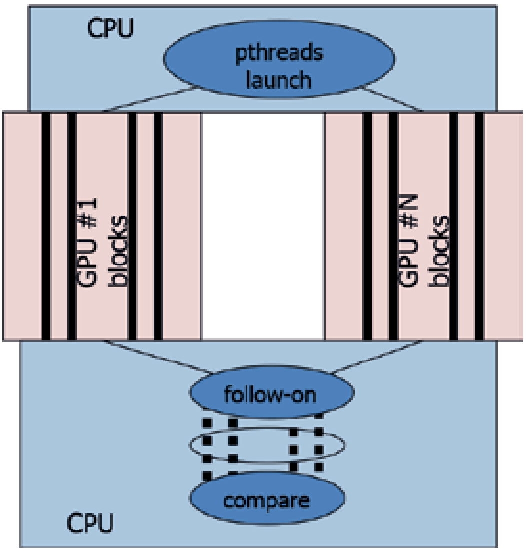


Figure 9. Diagnosis of GPU blocks.

	Tesla	Fermi	Kepler	Maxwell
SM	30	16	8	16
CUDA cores/SM	8	32	32	192
DP cores/SM	1	—	64	4
SFUs/SM	1	4	32	32
Load/Store/SM	—	16	32	32

Table 2. Nvidia GPGPU micro-architecture specifications.

consists of a number of threads that compute the chaotic-detection maps. The chaotic-detector maps computed by each GPU thread consist of the following expanded versions of the steps described in Section 3.1.

(i) The GPU SM id, thread id, warp id, and warp lane numbers are obtained, and the global device memory is allocated to thread T as an array $GM_{G:T}[.]$ of specified size S_T.

(iv-a) A set of variables X_K corresponding to the FP and integer portions of the CUDA core, DP core, and SFU are retrieved from each thread.

(iv-b) The variables X_K are stored and retrieved from each element of the assigned partition of the global memory such that the thread T executes the following code.

```
for $j = 1, \ldots, S_T$.
    $GM_{G:T}[i, j] = X_K$;
    $X_K = GM_{G:T}[i, j]$;
```

(v) N_2 iterations of the chaotic-detection map are computed with starting state X_K to obtain final state X_E;

The GPU diagnosis code is written as a CUDA kernel which is launched from CPU onto GPU blocks concurrently with pthreads on CPU cores. This diagnosis code can be significantly simplified for simpler diagnosis tasks, for example, checking PCI bus transfer errors between CPU and GPU, by simply writing and reading back the global memory arrays.

3.5. GPU diagnosis using OpenCL kernels

AMD GPGPUs and APUs have transitioned to the graphics core now (GCN) Architecture with the GPGPUs and APUs containing multiple GCN compute units (CU). For example, the AMD Radeon HD7970 (Firepro W9000) architecture consists of 32 GCN CU's operating with a 975 MHz clock while the AMD A10-7850 K Kaveri APU architecture consists of 2 Steamroller CPU cores (3.7–4 GHz) and 8 GCN CUs (720 MHz) with a unified address space of up to 32GB. Each CU consists of 4 Vector Units (VU) and one Scalar Unit. Each VU in turn consists of 16 SIMD multi-precision ALUs and a register file. In the case of the Firepro W9000 this is a total of 2048 ALUs (32CUs × 4VUs × 16 ALUs) with a theoretical peak of up to 1 TF DP and 4 TF SP performance. The basic scheduling unit of work for a GCN CU consists of a wavefront which is 64 threads. Each

SIMD VU within a CU has its own program counter and instruction buffer which can contain up to 10 wavefront buffers. Four CUs currently share a 32 KB L1 instruction cache. At a given cycle the 4 VUs of one CU can be operating on different wavefronts with a given wavefront completing in 4 clock cycles. In effect, a single CU could have up to 4 × 10 wavefronts in flight. Associated with each GCN CU is a general purpose register file which consists of 4 independent slices, one for each VU. Each slice consists of 256 vector registers (vGPRs) shared across 10 wavefronts with each vGPR being 64 lanes of 32 bits allowing 64 SP or 32 DP values to be stored in each vector register at each cycle. For further details we refer to [35].

The diagnosis module for AMD GPUs is written in OpenCL with separate kernels for chaotic map fault detection of SP and DP FPUs, integer ALUS, extended math units, and register files. Since the diagnosis modules are written in OpenCL they also run on multi-core CPUs and GPUs from other vendors such as Nvidia. However, obtaining thread level information is much harder with OpenCL and requires the use of vendor specific analysis tools. Hence, the OpenCL chaotic map detection modules provide less quantitative information than the modules written with pthreads for Xeon Phi and CUDA for Nvidia GPGPUs. However, these diagnostic codes provide a good tool for fault detection on hybrid systems due to the cross-platform portabiity that OpenCL provides.

4. Experimental results

The diagnosis codes have been implemented in C using *float* and *double* datatypes based on the logistic and tent maps, and have been developed and tested in stages on the systems listed in **Table 1**. The *test modes* are represented as multi-core CPU (C), manycore processor (MC), single GPU (G), multiple GPUs (MG) or hybrid node with CPU and GPU (CG). The tests were carried out in C, MC, G, MG, and CG modes. In these systems, Xeon Phi's and GPUs are attached to CPUs over PCI bus, and are used as accelerators in all our systems, except in HP Z200 workstation where the GPU is used only for display. The following are the details of systems used in our code implementation and testing.

(C) **Multi-Core CPU:** Five different multi-core systems: 4-core Intel Xeon 2.67 GHz, 16-core AMD Opteron 2.3 GHz, 16-core Intel Xeon, 32-core Intel Xeon 2.7GHz, and 48-core AMD Opteron 2.29GHz.

(MC) **MIC Processors:** Intel Xeon Phi 3120P/A coprocessor.

(G) **Single-GPU:** Six cases: Quadro 600, Quadro K5000, Tesla T10, Tesla C1060, Tesla K20X, and AMD Firepro 9000 GPUs connected to CPU over PCI bus.

(MG) **Multiple-GPUs:** Two cases: 4-socket 48-core HP server with eight 8 Tesla T10 GPUs connected over four PCI busses and Apple MacBook Pro with an Intel HD Graphics 4000 GPU and a Nvidia GT 650 M GPU.

(CG) **Hybrid nodes:** AMD A10-7850 K Kaveri APU with 2 Steamroller CPU cores (3.7–4 GHz) and 8 GCN CUs (72 MHz)

Together these systems represent quite different software environments and architectures, and our diagnosis codes are compiled separately on them under Linux-like environments. But otherwise these codes are portable using C, CUDA, and OpenCL compilers with the pthreads libraries. The diagnosis test results are qualitatively quite similar across these systems. So, we present high-level summaries of our results with some representative traces, and briefly describe (somewhat) unexpected cases that may require additional considerations for developing application codes. For multi-core CPUs no errors were detected on these systems, and as expected the chaotic map outputs X_E's are identical for all cores. We simulated different faults to verify the functionality of diagnosis codes. When GPU are utilized, some interesting precision and emulation artifacts were observed in G and MG modes (Section 4.3).

4.1. Multi-core CPU diagnosis

Four different CPUs have been tested in C mode. The outputs X_E's of all chaotic-detection maps are identical in all these systems, and the results for the 4-core CPU are shown below using the tent and logistic maps ($N_1 = 20, N_2 = 0$ iterations with $X_0 = 0.2$):

Tent map:

```
Core 0: output: 0.165669 : 3E29A528
Core 1: output: 0.165669 : 3E29A528
Core 2: output: 0.165669 : 3E29A528
Core 3: output: 0.165669 : 3E29A528
Core 0: output: 0.165669 : follow_on: 0.919737
Core 1: output: 0.165669 : follow_on: 0.919737
Core 2: output: 0.165669 : follow_on: 0.919737
Core 3: output: 0.165669 : follow_on: 0.919737
```

Logistic map:

```
Core 0: output: 0.787269 : 3F498A78
Core 1: output: 0.787269 : 3F498A78
Core 2: output: 0.787269 : 3F498A78
Core 3: output: 0.787269 : 3F498A78
Core 0: output: 0.787269 : follow_on: 0.062074
Core 1: output: 0.787269 : follow_on: 0.062074
Core 2: output: 0.787269 : follow_on: 0.062074
Core 3: output: 0.787269 : follow_on: 0.062074
```

Outputs from all four threads from the individual cores are identical indicating no errors. The output consists of two parts: first part shows the chaotic-detection map outputs X_E's from the individual cores, and the second part shows the outputs of follow-on chaotic map X_F's. In the first part, the state of chaotic-detection map X_E is printed in C float format in the first column, and in hexadecimal representation in the second column. For the follow-on chaotic map, X_E and X_F are shown in the first and second columns, respectively. These outputs are the same in all four multi-core CPUs and three systems tested.

Since there are no errors detected on the CPU cores above, we simulated four types of errors:

a. We add a small quantity to X_K during the arithmetic operations for thread 0 to simulate ALU errors.

b. We simulate stuck-at memory errors by clamping X_K to a fixed value 0.000001 during the store and retrieve operation for thread 1.

c. We simulate data path errors by replacing X_K by a randomly generated number for a thread 2 during the store and retrieve operation.

d. We flip the outcome of the logical operation in one iteration in X_K computation for thread 3.

The faults (a)–(c) are applicable to both logistic and tent maps, and fault (d) is applicable only to the tent map.

The output for 4-core processor with four faults simulated on different cores, namely type (a) through (d) on cores 0 through 3, respectively, are shown below for the tent map:

Diagnosis summary:

```
Core 0: output: 0.000370 : 39C21000
Core 1: output: 0.000001 : 358637BD
Core 2: output: 0.010000 : 3C23D70A
Core 3: output: 0.000510 : 3A05A000
Core 0: output: 0.000370 : follow_on: 0.117106
Core 1: output: 0.000001 : follow_on: 0.960860
Core 2: output: 0.010000 : follow_on: 0.795249
Core 3: output: 0.000510 : follow_on: 0.045228
```

The outputs from these threads are different from those above indicating an error during the execution of each of them. Additionally, the final outputs of each of these chaotic trajectories are different from each other indicating different types of faults. The global memory $GM[.,,]$ is allocated prior to invoking the threads on the processors cores, and local memory $LS[.]$ is allocated within the thread assigned to a particular core. Since memory movements are carried out by all cores between their local memory and global memory, significant portion of the memory data paths are exercised by the diagnosis code so that major errors in memory bus and interconnect can be detected. An exhaustive coverage of all memory data paths would require extensions of this method such as explicit placement of processes and their memory near the cores, which may be achieved using NUMA tools.

4.2. Xeon phi diagnosis

Below we show a partial output from running the chaotic tent map detection on the Xeon Phi. For lack of space we have only shown the outputs for a few cores and for the SP and DP ALU calculations. Similar fields are outputted for the EMU, the integer ALU the ×87 math coprocessor, and the vector registers for each hardware thread. In addition to the partial summary shown a detailed output is written to file.

```
Diagnosis summary:

Number of cores detected = 228

  Core 003: SP-ALU : 3F7E86A2, DP-ALU : 3FD711AFCA21B76F

  Core 000: SP-ALU : 3F7E86A2, DP-ALU : 3FD711AFCA21B76F

  Core 001: SP-ALU : 3F7E86A2, DP-ALU : 3FD711AFCA21B76F

  Core 002: SP-ALU : 3F7E86A2, DP-ALU : 3FD711AFCA21B76F

  …

  Core 226: SP-ALU : 3F7E86A2, DP-ALU : 3FD711AFCA21B76F

  Core 227: SP-ALU : 3F7E86A2, DP-ALU : 3FD711AFCA21B76F
```

Outputs from the different threads are identical indicating no errors for the various components tested.

4.3. GPU diagnosis

Four GPUs have been tested in G mode and one is tested in MG mode with pthreads. A single thread is used on each block to compute the chaotic-detection map. The outputs of chaotic-detection maps are identical in all these cases, analogous to the CPU case, when the chaotic-map output is computed without adding the index value to X_F; also, the results are the same as the CPU case when faults were simulated. Recall that to keep track of outputs from individual blocks, the index (block number) was added to X_E, which was then subtracted on CPU to compute X_F. This specific combination of operations involving integers and fractions yielded non-uniform precisions among different blocks of the same GPU. We now briefly describe the details of such cases, and such effects have been observed on all GPUs in **Table 1**. The outputs of diagnosis codes from GPU of Titan using the logistic map are shown below in a condensed form so that only lines corresponding to blocks with different outputs are shown (when no faults are simulated):

Chaotic detection map:

```
block_x[0] = 0.682320 <-> 3F2EAC8E
…
block_x[2] = 2.682321 <-> 3F2EAC90
…
block_x[16] = 16.682320 <-> 3F2EAC80
block_x[17] = 17.682320 <-> 3F2EAC80
```

Follow-on chaotic map

```
block_x[0] = 0.682320 <-> 0.860477
…
block_x[2] = 2.682321 <-> 0.000000
…
block_x[16] = 16.682320 <-> 0.671719
block_x[17] = 17.682320 <-> 0.671719
```

Follow-on linear map

```
block_x[0] = 0.682320 <-> 0.000016
…
block_x[17]=17.682320 <-> 0.000016
```

The chaotic detection map outputs X_E's (fractional part in the first column) are not the identical across the blocks, and the differences are significant enough to be noticed when printed under C float format. The differences are more clearly seen in hexadecimal format; here the block number has been subtracted from the first column number. The outputs of the follow-on chaotic-map X_F's more clearly show significant deviations as these small precision differences in X_E's are non-linearly amplified. As an additional step, we also computed the outputs of a follow-on linear-map, $M(X) = X + \delta$, which shows that these differences are inconsequential, and it also shows that some linear maps do not provide the needed detection capability.

The outputs from Quadro 600 GPU are shown below using the tent map, wherein the results are qualitatively similar to Titan K20X GPU but the details differ.

Chaotic detection map:

```
block_x[0] = 0.170387 <-> 3E2E79D8
…
block_x[2] = 2.170387 <-> 3E2E79E0
```

```
…

block_x[8] = 8.170386 <-> 3E2E79C0

…

block_x[16] = 16.170387 <-> 3E2E7A00
```

Follow-on chaotic map:

```
block_x[0] = 0.170387 <-> 0.313038

…

block_x[2] = 2.170387 <-> 0.793185

…

block_x[8] = 8.170386 <-> 0.459723

…

block_x[16] = 16.170387 <-> 0.903821
```

Follow-on linear map:

```
block_x[0] = 0.170387 <-> 0.000041

…

block_x[16] = 16.170387 <-> 0.000041
```

The transition points of X_E are different in this case compared to the logistic map case, and in both cases they varied based on the number of blocks used by the CUDA kernel. But, these outputs are the same across all four GPUs used in our tests. These artifacts are related to the real number precision on GPU blocks. Similar precision effects also occur in CPU cores, and the application codes account for them in some cases by using double precision computations. Except on K20X GPUs, only single precision is supported on GPUs used in our tests. But, even when the same single precision (C float) operations are used, these precision effects are different between CPU cores and GPU blocks. To compare to CPU tests, we added the core number to X_E and subtracted it on host core, and no differences were found in X_E‘s using C float print; the largest number of cores we tested is 48, and such precision effects may indeed manifest when larger numbers are added. Consequently, if not adequately accounted for, these precision differences could lead to potentially unpredictable results in certain non-linear computations, particularly if automated tools are used to convert CPU codes to CPU-GPU hybrid systems.

4.4. Hybrid systems diagnosis

For the purposes of this subsection hybrid systems are considered as consisting of a mixture of CPU and GPU cores. Results are presented for three such systems using OpenCL kernels to perform the chaotic-map detection. The simplest system is the platform on a Macbook Pro which OpenCL detects as a single OpenCL platform with three different compute devices:

Intel(R) Core(TM) i7-3720QM CPU @ 2.60GHz, an Intel HD Graphics 4000 device, and an NVIDIA GeForce GT 650 M. Below are representative partial outputs for the single and double precision FPU tent map computations. Not show are computations for the integer ALU and EMU. Diagnosis Summary:

```
Device 0, SP-ALU : 3F1B0A10, DP-ALU : 3FE9B1B7A9B71338

Device 1, SP-ALU : 3F1B0A10, DP-ALU : FF800000FF800000

Device 2, SP-ALU : 3F1B0A10, DP-ALU : 3FE9B1B7A9B71338
```

Note that the DP ALU results for device 1 (Intel HD Graphics 4000) differ from the other devices as device 1 does not possess double precision capability.

The second hybrid system we consider is a node that OpenCL detects as consisting of two platforms with one and two compute devices, respectively. Platform 0 is an NVIDIA platform with a NVIDIA Tesla K20c device. Platform 1 is an AMD platform with device 0 an AMD Tahiti device (Firepro 9000) and device 1 an Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz. Below are representative partial outputs for the double precision FPU tent map computations. Not shown are calculations for the SP ALU, integer ALU, and EMU. All systems return the same result in the absence of errors. Diagnosis Summary:

```
Platform 0, device 0, DP-ALU : 3FE9B1B7A9B71338

Platform 1, device 0, DP-ALU : 3FE9B1B7A9B71338

Platform 1, device 1, DP-ALU : 3FE9B1B7A9B71338
```

The third system considered is a node with an AMD A10-7850 K APU. OpenCL identifies it as one platform with two devices. Device 0 is identified as AMD Spectre which consists of 8 GCN cores and Device 1 is identified as AMD A10-7850 K APU which consists of 4 CPU cores. Below are representative partial outputs for the single and double precision FPU tent map computations. Not show are computations for the integer ALU and EMU.

Diagnosis summary:

```
Device 0, SP-ALU : 3F1B0A10, DP-ALU : 3FE9B1B7A9B71338

Device 1, SP-ALU : 3F1B0A10, DP-ALU : 3FE9B1B7A9B71338
```

4.5. Operational artifacts

Our diagnosis codes were originally developed for low-level hardware faults, such as in ALU and interconnects. During the tests, however, they detected certain artifacts, which could lead to inconsistencies and/or errors in some computations if not adequately accounted for:

a. *Tardy computations*: In some systems, GPUs are emulated on the nodes, particularly if they housed them previously, and the emulated codes run sequentially on CPUs and lead to tardy computations. Our codes explicitly check for physical GPUs, and detected such emulations on nodes. Also, in SN-MG tests, we explicitly scheduled kernels on devices with numbers outside the eight GPUs, and the computations on them were completed sometimes with incorrect results. These tests call for suitable device checks by application codes.

b. *Data transfer errors*: When CUDA kernels are launched and the results are gathered using MPI, certain elements received zero values. The occurrence of these errors was random but the zero elements always appeared in the blocks whose number matched the node.

These results provide information of interest to systems operations and application development.

5. Conclusions

We described a method to quickly detect certain faults in hybrid computing systems consisting of multi-core processors and accelerators by utilizing chaotic map computations. Our implementation is based on pthreads for multi-core CPUs and MICs, and CUDA C and OpenCL kernels for GPUs. We presented experimental diagnosis results on five multi-core CPUs, one MIC, seven GPUs and three hybrid systems. Since the original systems are not faulty, we simulated certain faults in arithmetic operations, local and global memory elements, data paths, and processor interconnects, which were detected. In addition, these codes identified artifacts of non-uniform precisions of GPU blocks and tardy hybrid computations, which could be of interest to non-linear computations.

Deeper investigations are needed to characterize the class of faults detected by a given set of chaotic maps, augmentation and data movement operations. While the logistic and tent maps used in our tests was able to detect the simulated faults, it would be interesting to study different chaotic maps whose Lyapunov exponents closely match the specific faults to minimize the detection times. More generally, it would be interesting to study the class of diagnosis algorithms that are optimal for a given class of faults. In terms of implementations, it would be interesting to explore finer control of memory allocations and data paths in movement operations using NUMA to further refine the diagnoses. In addition, assignment operations under OpenACC, SHMEM and PGAS involve data movements across complex data paths, and it would be interesting to explore the faults that can be detected by using them for content-preserving data movement operations. The proposed chaotic maps can be embedded into applications to track their execution paths so that faults can be detected during their execution. More generally, the fault diagnosis codes could be an integral part of overall ecosystems needed for resilient computations, and it would be of interest to co-develop them.

Acknowledgements

This work is supported in part by the United States Department of Defense and used resources of the Computational Research and Development Programs, and is also supported in part by Applied Mathematics Program, Office of Advanced Computing Research, Department of

Energy at Oak Ridge National Laboratory managed by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725. We acknowledge the support of Oak Ridge Leadership Computing Facility (OLCF) in making Lens, Titan and Chester systems available, and the information and guidance from Don Maxwell of OLCF that contributed to implementation and testing.

Author details

Nageswara S. V. Rao[1]* and Bobby Philip[2]

*Address all correspondence to: raons@ornl.gov

1 Oak Ridge National Laboratory, Oak Ridge, TN, USA

2 Los Alamos National Laboratory, Los Alamos, NM, USA

References

[1] Vetter JS, editor. Contemporary High Performance Computing: From Petascale toward Exascale. Boca Raton, Florida, USA: Chapman and Hall/CRC Press; 2013

[2] Cappello F. Fault tolerance in petascale/exascale systems: Current knowledge, challenges and research opportunities. Journal of High Performance Computing Applications. 2009; **23**(3):212-226

[3] Dongarra J, Beckman P, et al. The international exascale software roadmap. International Journal of High Performance Computer Applications. 2011;**25**(1):3-60

[4] Cappello F, Geist A, Gropp B, Kale S, Kramer B, Snir M. Towards exascale resilience. Journal of High Performance Computing Applications. 2011;**23**(4):374-388

[5] Li M, Ramachandran P, Sahoo SK, Adve SV, Adve VS, Zhou Y. Understanding the propagation of hard errors to software and implications for resilient system design. In: Architectural Support for Programming Languages and Operating Systems (ASPLOS). 2008

[6] Erez M, Jayasena N, Knight TJ, Dally WJ. Fault tolerance techniques for the merrimac streaming supercomputer. In: International Conference for High Performance Computing, Networking, Storage and Analysis (SC); 2005

[7] Li D, Chen Z, Wu P, Vetter JS. Rethinking algorithm-based fault tolerance with a cooperative software-hardware approach. In: ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis; 2013

[8] Marotto FR. Snap-back repellers imply chaos in r^n. Journal of Mathematical Analysis and Applications. 1978;**63**:199-223

[9] NVIDIA. NVIDIA GeForce GTX 200 GPU Architectural Overview. Santa Clara, CA, USA: NVIDIA Corporation; 2008

[10] NVIDIA. NVIDIAs Next Generation CUDA Compute Architecture. Santa Clara, CA, USA: NVIDIA Corporation, Fermi; 2009

[11] NVIDIA. NVIDIAs Next Generation CUDA Compute Architecture. Kepler GK110/210. Santa Clara, CA, USA: NVIDIA Corporation; 2012

[12] NVIDIA. NVIDIA GeForce GTX 980 Featuring Maxwell, The Most Advanced GPU Ever Made. Santa Clara, CA, USA: NVIDIA Corporation; 2014

[13] Parker TS, Chua LO. Chaos: A tutorial for engineers. Proceedings of the IEEE. 1987;**75**(8): 982-1008

[14] Rahman R. Intel Xeon Phi Coprocessor Architecture and Tools: The Guide for Application Developers. 1st ed. Berkely, CA, USA: Apress; 2013

[15] Rao NSV. Chaotic-identity maps for robustness estimation of exascale computations. In: 2nd Workshop on Fault-Tolerance for HPC at Extreme Scale (FTXS 2012); 2012

[16] Rao NSV. Fault detection in multi-core processors using chaotic maps. In: 3rd Workshop on Fault-Tolerance for HPC at Extreme Scale (FTXS 2013); 2013

[17] Rao NSV. On undecidability aspects of resilient computations and implications to exascale. In: Resilience 2014: Seventh Workshop on Resiliency in High Performance Computing with Clouds, Grids, and Clusters; 2014

[18] Sahoo SK, Li M-L, Ramachandran P, Adve SV, Adve VS, Zhou Y. Using likely program invariants to detect hardware errors. In: International Conference on Dependable Systems and Networks; 2008

[19] HPL—A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers. Available from: http://www.netlib.org/benchmark/hpl

[20] Wiggins S. Introduction to Applied Nonlinear Dynamical Systems and Chaos. New York: Springer-Verlag; 1990

[21] Alligood KT, Sauer TD, York JA. Chaos: An Introduction to Dynamical Systems. New York: Springer-Verlag; 1997

[22] Fujiwara H, Toida S. The complexity of fault detection problems for combinational logic circuits. IEEE Transactions on Computers. 1982;**C-31**(6):553-560

[23] Chen Z. Online-abft: An online algorithm based fault tolerance scheme for soft error detection in iterative methods. In: ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming; 2013

[24] Davies T, Chen X. Correcting soft errors online in lu factorization. In: Symposium on High-Performance Parallel and Distributed Computing; 2013

[25] Huang Y, Kintala C. Software fault tolerance of the application layer. In: Lyu MR, editor. Software Fault Tolerance. New York, NY, USA: John Wiley & Sons, Inc; 1995. pp. 231-248

[26] Jia Y, Luszczek P, Bosilca G, Dongarra J. CPU-GPU hybrid bidiagonal reduction with soft error resilience. In: Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (ScalA); 2013

[27] Carbin M, Misailovic S, Rinard MC. Verifying quantitaive realiability for programs that execute on unreliable hardware. In: Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA); 2013

[28] de Kruijif M, Nomura S, Sankaralingam K. Relax: An architectural framework for software recovery of hardware faults. In: International Symposium on Computer Architecture (ISCA); 2010

[29] Gao J, Cao Y, Tung WW, Hu J. Multiscale Analysis of Complex Time Series. Hoboken, NJ, USA: John Wiley and Sons; 2007

[30] Hilborn RC. Chaos and Nonlinear Dynamics. Oxford, England, UK: Oxford University Press; 1994

[31] Intel. Intel Xeon Phi Coprocessor Instruction Set Architecture Reference Manual. Santa Clara, CA, USA: Intel Corporation; 2012

[32] Intel. Intel Xeon Phi X100 Family Coprocessor—The Architecture. Santa Clara, CA, USA: Intel Corporation; 2012

[33] Jeffers J, Reinders J. Intel Xeon Phi Coprocessor Architecture High Performance Programming. 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2013

[34] Kirk DB, Hwu WW. Progammng Massively Prallel Processors: A Hands-on Approach. Second ed. Morgan Kaufman Pub; 2013

[35] AMD. AMD Graphics Core Next (GCN) Architecture. Santa Clara, CA, USA: Advanced Micro Devices, Inc; 2012

Section 2

Power Systems

Chapter 2

Preventive Maintenance and Fault Detection for Wind Turbine Generators Using a Statistical Model

Ian Kuiler, Marco Adonis and Atanda Raji

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.80071

Abstract

Vigilant fault diagnosis and preventive maintenance has the potential to significantly decrease costs associated with wind generators. As wind energy continues the upward growth in technology and continued worldwide adoption and implementation, the application of fault diagnosis techniques will become more imperative. Fault diagnosis and preventive maintenance techniques for wind turbine generators are still at an early stage compared to matured strategies used for generators in conventional power plants. The cost of wind energy can be further reduced if failures are predicted in advance of a major structural failure, which leads to less unplanned maintenance. High maintenance cost of wind turbines means that predictive strategies like fault diagnosis and preventive maintenance techniques are necessary to manage life cycle costs of critical components. Squirrel-Cage Induction Generators (SCIG) are the prevailing generator type and are more robust and cheaper to manufacturer compared to other generator types used in wind turbines. A statistical model was developed using SCADA data to estimate the relationships between winding temperatures and other variables. Predicting faults in stator windings are challenging because the unhealthy condition rapidly evolves into a functional failure.

Keywords: fault diagnosis, preventative maintenance, wind turbine, electrical generator, statistical model

1. Introduction

Wind energy has evolved into a mature, cost effective and sustainable power technology. The sizes of wind turbines are growing on a continuous basis and new topologies allow for better integration into electricity grids. Power electronics development has provided the

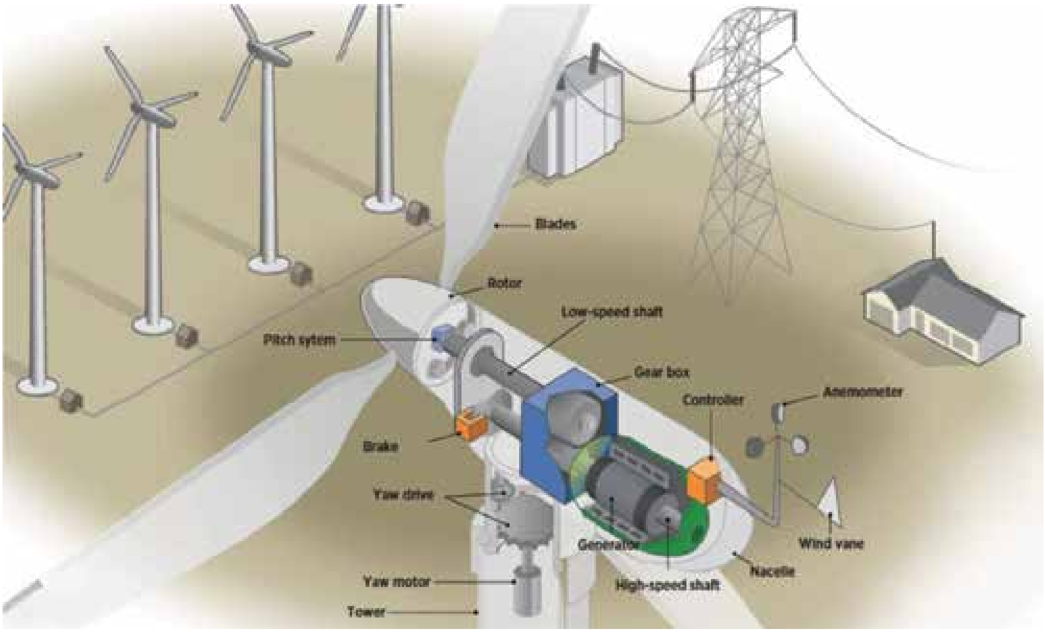


Figure 1. Wind turbine components [1].

functionality of variable speed operation, which is more energy efficient. A wind turbine typically comprises 8000 parts or more with the blades, rotor, main bearing, drivetrain and power module its major components. **Figure 1** depicts the typical components of a wind turbine. A major component of the power module is the electrical generator. Squirrel-Cage Induction Generators (SCIG) are currently the most common electrical generator type used in wind turbines, because these are robust and cheaper to manufacturer compared to other generator types. As a complex power system it is important to understand how failures in wind turbines occur despite its current level of maturity. High reliability and availability is thus expected over a typical 20-year design life.

2. Wind energy overview

2.1. Wind energy evolution

Wind energy adoption has seen year-on-year continued growth and implementation. The global installed wind energy capacity is illustrated in **Figure 2**.

The operating principle of all wind turbines make use of either aerodynamic lift or aerodynamic drag forces. Aerodynamic lift forces are perpendicular to the direction of the wind whereas drag forces are in the same direction. Modern day wind turbines are mainly designed to use aerodynamic lift forces where the rotor blades are turned into the direction of the wind. The perpendicular lift force produces the required driving torque via the leverage of the rotor.

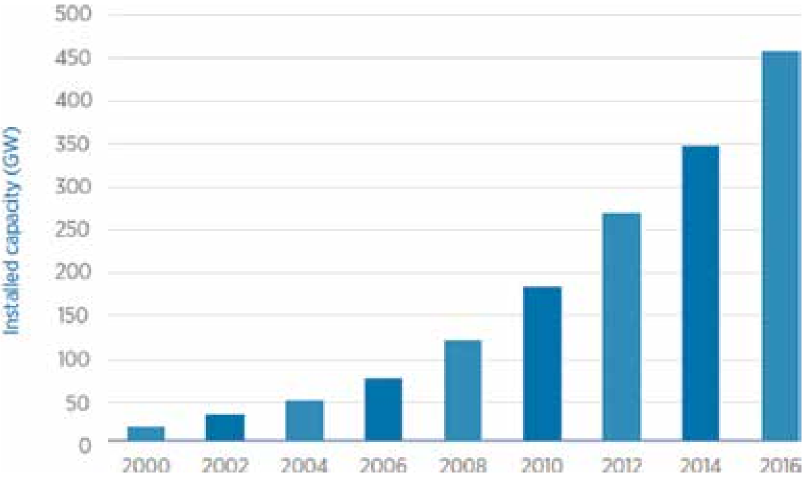


Figure 2. Global wind energy installed capacity [2].

Only wind turbines operating on aerodynamic lift will be discussed here and these are classified in accordance to the direction of the rotating axis i.e. horizontal axis wind turbines (HAWTs).

2.2. Wind energy cost

Wind energy has reached commercial maturity remarkably fast and has seen its cost dropped significantly to such levels that it's now cost competitive with coal power generation [2]. For any power generation technology, the cost of production is variable and influenced by technology maturity, operating conditions, location and the capacity rating of the plant [3]. The LCOE for wind energy is affected mainly by the following factors [4]:

- Operation and Maintenance (O&M) costs;
- Annual energy production (AEP);
- Capital costs;
- Financing costs.

Figure 3 indicates the capital cost breakdown of all the major wind installation of a typical onshore wind turbine and it is evident that the major costs are related to the turbine itself. LCOE can be reduced if wind turbine manufacturers enhance turbine technology so that a variety of designs are available for different wind resource conditions. This can be achieved through larger rotors, improved blade aerodynamics and taller towers [4].

The capacity factor (CF) indicates how frequently the wind turbine was able to produce power at rated or name plate capacity over a given period (normally a year). Capacity factors for onshore wind turbines fall in the range between 30 and 35% [4]. This figure varies

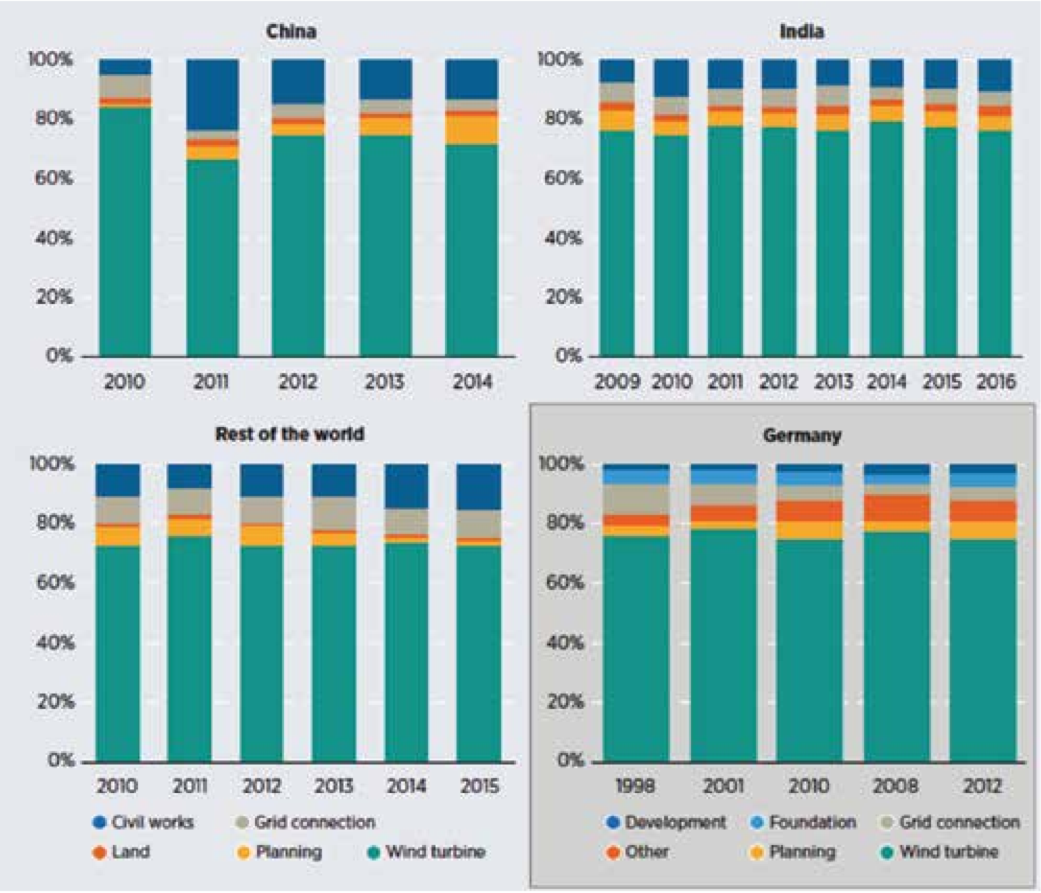


Figure 3. Cost breakdown of onshore wind farms [2].

considerably depending on turbine design and the local wind resource. In conventional power generation technologies the AEP is generally proportional to the generator size. However in a wind turbine the rotor swept area can have a bigger influence than the generator size on the power generation capability [5].

Therefore the relationship between the rotor swept area and generator size can influence capacity factors of wind turbines. In other words a wind turbine with a specific rotor swept area connected to two different size generators will have different capacity factors. The smaller size generator will operate at a higher capacity factor compared to the bigger size generator with the same wind conditions. Wind turbine manufacturers should therefore optimise this relationship for specific site conditions and grid integration requirements to ensure the lowest possible costs. O&M costs of wind turbines vary over the lifespan of the plant and escalate with age as the risks of failure of the equipment increase. The O&M costs of wind turbines have reduced considerable over the last 30 years and accounts between 20 and 30% of the total life cycle costs for onshore projects [2]. O&M costs for offshore wind projects are higher because of the severe operating conditions in the sea, access to site, complex maintenance tasks and transmission infrastructure costs. The costs for onshore wind projects are approximately USD 30-60/MWh versus USD 71-155/MWh for offshore projects [6].

2.3. Speed characteristics of wind turbines

Wind turbines can rotate at a fixed speed where the optimum energy conversion takes place at a specific wind speed or at variable speed which has a more complicated electrical design [7] but is efficient over a wind speed range. The fixed speed of the wind turbine technology depends on the gearbox ratio, frequency of the grid and the electrical generator design characteristics [8]. From 1980 to early 1990s all wind turbines used for large scale power generation was fix speed and used gearboxes.

Fixed speed wind turbines are rugged, cost effective to build but experience higher power fluctuations as a result of the constant generator speed against varying wind speeds [7]. These turbines unfortunately draw large reactive power from the grid which are compensated for by installing power factor correction capacitors. The disadvantage of power factor correction capacitors is power quality problems like harmonic resonance on the grid [8].

Variable wind speed turbines are designed to reduce mechanical stresses, maximise wind energy capture and provide smoother output power which is more suited to the grid. This technology became popular in the 1990s at the same time when advances in power electronics, reactive power control, variable speed induction generators and synchronous generator systems happened [9].

By connecting the electrical generator via a power electronics system to the grid, the wind turbine speed can be adjusted. Harmonic currents from the power electronics systems in variable speed wind turbines also cause power quality problems. Associated transient voltage peaks of 100 times more than the expected values between windings cause insulation damage of windings and ultimately failure of the machine [10].

For a certain wind resource with specific Weibull distribution parameters, it was shown that additional annual energy captured by a variable speed turbine was 2.3% more than a similar rated fixed speed turbine. The additional costs of a variable speed wind turbine compared to a fixed speed wind turbine of the same rating at a given location are off-set by its ability to capture more energy in the wind [10].

The study in [10] revealed that a variable speed wind turbine produces more power than the fixed speed turbine of the same rating. Although the difference might appear small, the amount of power generated over the life cycle of the wind turbine which is typically 20 years can deliver substantial generation profit.

Power regulation is normally done by pitching the rotor blades, stall control or a combination of the two in order to avoid overloading the wind turbine. The aerodynamic forces acting on the rotor and the output power of the turbine are reduced during high wind speeds. Variable speed wind turbines in conjunction with dynamic blade pitch for power and load control is considered as the accepted industry standard for most modern wind turbines.

2.4. Wind turbine classes

Factors such as the average yearly wind speed magnitude, wind turbulence and severe gusts speeds, determine if a wind turbine design is suited for safe operation at a particular site. The International Electrotechnical Commission (IEC) standard IEC 61400-1, stipulates the different wind turbine classes based on aerodynamic loading [11]. Wind turbines classified as low wind "Class IV" i.e. S111 according to IEC 61400-1 are now becoming feasible to enter the power generation market [11].

Wind classes I, II and III can be equated to high, medium and low wind sites in general. Locations with low wind resources are suited for wind turbines designed with bigger rotors and higher towers to balance energy conversion and costs. These wind turbines types are largely coupled to smaller drivetrain and power generating units to increase their effectiveness in these less promising wind conditions. Medium and low wind turbines have become more popular than high wind turbines with Asia leading the international market.

3. Wind turbines generator types

The electrical generator in the wind turbine converts the mechanical energy from the turbine rotor into electrical energy which is supplied to the grid. In conventional power systems where synchronous generators are used, power is produced at constant speed. Applying these generating systems in wind energy is a challenge because of the variable nature of the resource.

Induction generators also known as asynchronous generators because they do not rotate at a fixed speed are the most commonly used electrical generator in WECs today. The application of induction generators in the power industry is limited compared to induction motors, which are seen as the workhorses in power systems consuming approximately 33% of global generated electricity. There are several advantages that make induction generators suitable for wind energy technologies as mentioned by Das et al. [12].

Induction generators are classified according to their rotor structure, which is, squirrel cage and wound rotor types. The stator designs of both induction machines are the same. The term power converter in the following paragraphs refers to all power electronic systems such as soft-starters, inverters, rectifiers or frequency converters.

3.1. The squirrel cage induction generator

SCIGs are used in fixed speed or variable speed wind turbine concepts. The SCIG stator is connected to the grid via a power transformer and a power converter is used to reduce the inrush current. The function of the capacitor bank is to reduce the reactive power consumption and support the generator voltage. This configuration is also known as the Danish concept and the first generation was directly connected to the grid without any power converters Technology developments and subsequent reduction in power electronics costs have been main drivers for the use of SCIGs in variable speed wind turbines. The generator is connected to the grid via a full rated converter, which controls the stator current instead. This configuration has full control of real and reactive power and operates across the full speed range.

The size of the generator is more compact and lighter compared to other full converter designs. This type configuration is predominantly used by Siemens Wind Power which has a 4.1% global market share [11]. According to [11] North America has an installed capacity of 1.5 GW, the rest of the world 0.98 GW excluding European and Asian markets.

The power quality of SCIGs at low and high wind speeds are better compared to wound rotor induction generators (WRIGs) while the latter produce less harmonics near synchronous speed [76]. Other attributes, which make SCIGs desirable over WRIGs, are:

- Better grid stability because of the larger converter;
- No brushes or slip ring maintenance as well as reduced losses;
- Robust rotors which can provide better electrical and mechanical performance;
- It is cost effective and readily available.

The converter in this configuration needs to be sized to the full capacity of the generator, which makes it very expensive. The harmonic filters are also rated at full converter capacity which is costly and difficult to design [13]. The performance of the converter has to be very good over the entire power range to ensure optimum efficiency and generation capacity.

3.2. Synchronous generators

Synchronous generators are matured technologies in fossil fuels and nuclear power systems and produce grid power at constant speeds. Their robustness and ability to control grid voltage by adjusting the rotor excitation make them ideal for power systems. This is particularly important during grid problems like faults where the generator is to remain connected to the grid and support the grid voltage through reactive power control. Because of these attributes synchronous generators are now being used in WECs and their rotors can be separately excited or make use of permanent magnets [13].

For a synchronous generator the absence of slip rings, gearbox and external excitation reduce the overall losses and the full rated power converter maintains its flexibility. The full rated converter and magnetic material costs make this concept very expensive but energy efficiency is improved [13]. Different permanent magnet synchronous generators designs are described and analysed by [14].

4. Failures in wind turbines

4.1. Wind turbine failures overview

Failures in wind turbines can result from various sources including poor quality, inferior design and manufacturing standards, construction and erection deficiencies, local operating conditions, transmission system design and general maintenance [15]. Mechanical failures occur most often, gearbox failures cause the longest downtimes and failure rates above one failure per turbine annually is still common [16]. The failure rate of the majority of wind turbine components or systems increase as designs move away from well-established designs towards new concepts, which are less, matured. A similar observation was made when the wind turbine generator rating increases from small to large [17]. In a study of about 800 wind turbines it was established that the availability was over 90% for the majority of turbines irrespective of size [15]. This study also showed that the difference between availability figures amongst major wind turbine manufacturers were small. The primary course of failures is due to wear out as the hazard rate increases during the last phase of component design life [17]. The authors in [16] concluded that gearbox failures cause the longest downtimes and that the average downtime reduced as technologies improved. The failure rates and downtime of subsystems during a survey done on more than 1500 wind turbines in Germany over a 15 year period show generator failures represent approximately 4% of the total number of failures in the wind turbines.

4.2. Generator failures

The major cause of failure in electrical machines irrespective of their applications is related to bearings and windings. The following components are responsible for the majority failures in wind generators using induction generators [18]:

- Bearings;
- Winding failures in both the stator and rotor;
- Rotor cages and leads;
- Slip rings;
- Magnetic wedges in the stator;
- Cooling plant.

The size of the generator also influences which components fail as manufacturers try to optimise designs for various power requirements and wind conditions. The three major faults identified across various generator ratings are summarised in **Table 1** [18]. Failure modes 1–3 represent the major faults ranging from most dominant to less dominant failure modes.

Rotor winding problems in small to medium generators are caused by conductor and banding failures while stator winding problems are related to contamination and maintenance

Generator size	Failure mode 1	Failure mode 2	Failure mode 3
Small <1 MW	Rotor	Stator	Bearings
Medium 1–2 MW	Bearings	Collector rings	Rotor
Large >2 MW	Bearings	Stator	Stator wedge

Table 1. Major failure modes across different wind generator sizes.

issues. Failures of bearings, stator windings and rotor windings contribute more than 80% of the total failures in induction machines [18]. This translates to a failure distribution for bearings (41%), stator (37%), rotor (10%) and other faults (12%).

4.3. Stator failures

4.3.1. Stator windings failures

Main ageing mechanisms causing insulation failure of rotor and stator windings are thermal effects, vibration stresses, voltage spikes from the power converters and material degradation because of temperature changes. Environmental conditions can accelerate insulation degradation and moist operating conditions should be avoided. The occurrences of short circuits escalate with time and are caused by overheating; ageing and vibrations while open circuits result from termination problems or damaged windings. Voltage spikes caused by power converters in variable speed induction machines are also responsible for winding insulation failures. Because of very fast switching times in the PWM circuit, multiple reflected waves travel between the converter and the machine. Impedance differences between the output cable and the generator create these reflected waves which become more severe as the cable length increases and the switching frequency of the semiconductors increases [16]. The reflected waves occur at the front of the voltage wave and can reach magnitudes up to 2.5 kV for a generator rated at 690 V.

Winding insulation design requirements should comply with the following conditions as a minimum [19]:

- Design life and mean time between failure (MTBF) of 20,000 h under accelerated ageing tests conditions;
- Rated voltage capacity test plus 10–15% and then 2.5 kV peak—peak “withstand” voltage after the ageing test;
- Initial partial discharge voltage test higher than the maximum peak—peak voltage after ageing test.

4.3.2. Stator wedge failures

Conductive wedges are used to keep the stator windings in the core and secure it against mechanical forces and vibrations. It also improves efficiency, limits magnetic flux distortion, inrush currents and increases the thermal properties of the machine [19]. There are instances of exposed stator coils where the wedges came loose and fell out of the stator slots. **Figure 4** shows an example of this [18]. The rotating magnetic field is the main cause that stator wedges become loose and this can result in grounds faults and or damage to stator coils.

4.3.3. Bearing failures

Bearing failures contribute a significant amount towards wind generator failures and common causes are incorrect installation or misalignment as well as poor lubrication, overheating and mechanical breakage [15]. Bearing wear through normal ageing together with “indentation, smearing, surface distress, corrosion”, electric current flow and overloading can also

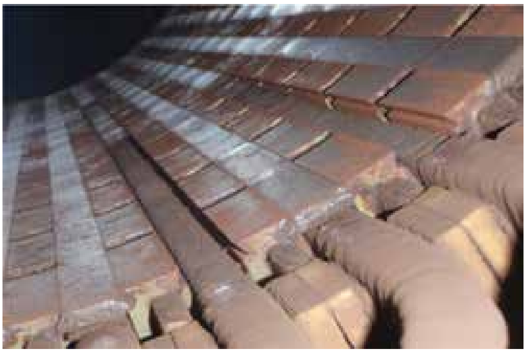


Figure 4. Missing stator slot wedges.

lead to bearing failure. It is recommended that maintenance practises comply with bearing lubrication schedules to reduce bearing failure rates. Damaged bearings can cause excessive vibrations of the rotor, which disturbs the uniform shape of the air gap between the stator and rotor. If not picked up these vibrations can cause contact between the stator and rotor, which will lead to catastrophic damage of both components.

5. Wind turbine maintenance

5.1. Maintenance strategies

Maintenance is the activity that assist production operations with optimum levels of availability, reliability and operability at the lowest cost. Maintenance strategies can be broadly classified into three main strategies namely breakdown maintenance, preventive maintenance and corrective maintenance.

Currently all three maintenance strategies or a combination of them are used in the wind industry depending on the age of the wind turbine. Breakdown maintenance is the typical "run to failure" approach, preventive maintenance is done before a problem leads to a failure and corrective maintenance is scheduled to rectify existing plant specific problems. Preventive maintenance is further classified as use-based or predictive maintenance and the former is performed at predetermined instances which is related to the age of the equipment or at certain expired calendar times [20]. Use-based maintenance can lead to over or under maintenance as resources are not optimally used [21].

Condition based maintenance has the capability to estimate the remaining useful life of equipment in order to implement the best maintenance strategy before failure occurs. Doing inspections or monitoring certain variables using sensors like temperature, voltage, current, noise or vibrations to determine the condition of the equipment can do it. The process of condition monitoring can be online or offline and is made up of three primary steps [22]:

- Data acquisition—gathering data that is pertinent to equipment health;
- Data processing—analytical verification, comprehension and refinement of collected data;

- Decision-making—deciding which maintenance strategy is ideal to ensure long term plant health at the lowest cost.

5.2. Condition monitoring techniques in wind turbines

The application of condition monitoring in WEC systems is ideal as concluded by [20]. Several condition monitoring techniques like oil analyses, vibration analysis, electrical effects, acoustic emissions, ultrasonic methods, radiographic inspections, strain measurements, thermography, temperature measurements, shock pulse method and equipment performance are used as discussed by [21]. Current wind turbine condition monitoring focus on critical equipment like the gearbox, generator and main bearing, which are high, cost components and cause long downtimes.

Vibration analysis is the most common condition monitoring method used in wind turbines although its ability to detect electrical faults could be limited. Its effectiveness in direct driven or other modern wind turbine concepts is also questionable. Probabilistic measures in addition to data received from sensors are required for a more precise determination of the equipment condition as the operating nature of wind turbines is stochastic.

5.3. Generator stator windings condition monitoring

Accurate condition monitoring techniques of stator winding faults are required as it is the second largest failure mechanism in generators. Shorted windings cause the most damage in the machine as it produces additional heat in the windings, which further reduce the design life of winding insulation material. These faults originate as undetected inter turn faults that gradually isolate multiple turns or when an arc exist between two points on a winding. Detection of inter turn winding faults is complex because the machine can still operate without any obvious fault signatures. These faults can rapidly evolve and cause complete failure of the winding and damage to the machine.

Temperature monitoring is considered as one of the oldest conditioning monitoring techniques and is commonly used in wind turbines to detect abnormalities in bearings and generator windings [22]. High stator winding temperatures under normal operating conditions is generally a sign of possible winding damage. Other factors such as high ambient temperatures or problems with the generator cooling have a similar effect. Insulation life is reduced by 50% for every 10°C increase in temperature as oxidation rates increase above certain temperature limits. Oxidation makes the insulation material fragile and some parts of the winding might experience delamination.

Majority of modern wind turbines are designed with condition monitoring systems, which incorporates a Supervisory Control, and Data Acquisition (SCADA) system. One of the functions of the SCADA system is to capture operating parameters from the wind turbine. Various mechanical and electrical sensors measure operating and performance data, which are recorded on a computer system for analysis. The SCADA data is typically recorded and stored by the computer system. Analysis of SCADA data for fault prognosis is seen as cost effective maintenance strategy although its data content does not reveal abnormalities in a clear and explicit manner.

Proper data analysis and modelling techniques are required to identify and understand component degradation. This will enhance component health predictions and guarantee the implementation of optimum maintenance strategies. According to [21] physical models depend on detailed understanding of failure modes whereas data driven models involve extensive data requirements to validate continuous degradation processes.

The application of SCADA data as a condition monitoring technique in the wind industry has become a prevalent research topic. These methods usually consist of various physical and statistical models of a particular system. Harmonics in line currents and magnetic flux, torque pulsations, reduced mean torque, high losses, abnormal winding temperatures and reduced efficiency are all indicators which highlight problems in induction machines [23]. The literature reveals that inter turn faults and asymmetries in the rotor or stator are the main focus of most condition monitoring techniques [23]. Electrical signature analyses of the stator parameters such as current, voltage and power under steady state operating conditions prove to be successful in sensing winding faults as well as other failure mechanisms.

6. System modelling and design

6.1. SCIG design parameters

The SCADA data was obtained from Siemens, the operator for the electrical utility, Eskom's Sere wind farm in the Western Cape, South Africa. This is a 100 MW wind farm with a total of 46 x 2.3 MW turbines. The SCIG in this study has the following design parameters as shown in **Table 2**.

6.2. Prediction model for stator winding temperatures

SCADA data from two wind turbines is used to model generator winding temperature between minimum and maximum output power which corresponds to 0–2.4 MW. Data for wind turbines (WTs) number 4 and number 38 were collected from June 2015 until October 2015. The maximum designed generator stator winding insulation temperature for the wind turbines is 155°C, which corresponds to a Class F rated insulation material.

Multiple linear regression analysis is a statistical method that estimates or model relationships between different variables that are linked in a nondeterministic way [24]. It uses more than one independent variable compared to linear regression, which has only one independent variable. The stator winding temperature prediction model is designed using Stepwise Regression (SR) in Microsoft Excel. The model output also highlights which variables have the biggest influence on stator winding temperature. Modelling of stator winding temperature in this study equates to the generator temperature.

SR performs multiple regressions that add or remove independent variables at each step based on performing a partial F-test on the new independent variable. The F-test calculates if different variables are mutually important and that their output has a significant effect on the dependent variable. It selects the independent variable with the highest correlation with the dependent variable initially, then adds or removes independent variables in the model based on calculating its F-test value, which should be higher or at least equal to the previous value.

SCIG 2.3 MW, 690 V, 50 Hz	
Rated output power	2.3 MW
Rated line to line voltage	690 V
Rated phase voltage	398.4 V
Rated stator current	2100 A
Rated stator frequency	50 Hz
Rated power factor	0.88
Rated rotor speed	1510 rpm
Synchronous speed	1500 rpm
Rated slip	−0.0069
Number of poles	4
Stator winding resistance, R_s	1.01 mΩ
Rotor winding resistance, R_r	1.3 mΩ
Stator leakage inductance, L_{ls}	0.093 mH
Rotor leakage inductance, L_{lr}	0.054 mH
Magnetising inductance, L_m	2.78 mH
Rated mechanical torque	16.313 kNm
Moment of inertia	63 kgm^3

Table 2. SCIG design parameters.

When there are two independent variables, the F-test value is calculated using [24]:

$$\mathrm{F}_1 = \frac{\mathrm{SS}_{\mathrm{R}}\left(\frac{\beta_1}{\beta_2, \beta_0}\right)}{MS_E(x_1, x_2)} \tag{1}$$

where F_1, F-statistic of independent variable x_1; SS_{R}, sum of squared residuals due to regression; MS_{E}, mean square error for the model containing x_1 and x_2; β_0, β_1, β_2, slope coefficients.

The following assumptions are made to establish how a linear regression model fits the data [24].

- The residuals should be uncorrelated random variables with a zero average and constant variance.
- The residuals should be normally distributed.
- The order of the model is correct and that the data being investigated has linear characteristics.

A linear regression model where the dependent variable Y is related to k regressor (independent) variables has the form [24]:

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \in \tag{2}$$

where Y, dependent variable; β_0, intercept; β_j, j = 0,1,…,k, regression coefficients; x, regressor variables; $\in$, random error term.

The model therefore provides an acceptable estimation of the dependent variable across certain ranges of the independent variables because the real relationship between them cannot be determined [24]. Regression coefficients represent the rate at which the dependent variable changes in relation to individual independent variables.

They are calculated in SR using the least squares method represented by the following matrix notations:

$$\begin{cases} y = \begin{pmatrix} y_1 \\ y_2 \\ .. \\ y_n \end{pmatrix}, \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ .. \\ \beta_n \end{pmatrix}, \in = \begin{pmatrix} \in_1 \\ \in_2 \\ .. \\ \in_n \end{pmatrix}, \beta = (X'X)^{-1} X' y \\ x = \begin{pmatrix} 1 & x_{11} & x_{12} & .. & x_{1k} \\ 1 & x_{21} & x_{22} & :: & x_{2k} \\ .. & .. & .. & :: & .. \\ 1 & x_{n1} & x_{n2} & & x_{nk} \end{pmatrix} \end{cases} \tag{3}$$

The calculation of the predicted value of y is obtained by [24]:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots\ldots + \beta_k x_k \tag{4}$$

6.3. Evaluating the adequacy of the model

The SR model needs to satisfy certain criteria to justify whether its linear function is sufficient to predict generator stator winding temperature over the proposed output power range of the wind turbines.

The following parameters are selected as variables in the SR model:

- **Ambient Temperature (AT)**

The AT refers to the outside temperature conditions. The outside air is used to cool the generator as well as the inside of the nacelle. This independent variable is labelled as "Mean Ambient Tmp" in the SR model.

- **Nacelle Temperature (NT)**

The temperature in the nacelle affects the generator operating conditions directly as well as other components. High nacelle temperatures cause the generator to run hotter which affects its performance. The nacelle temperature is not regulated. This independent variable is labelled as "Mean Nacelle Tmp" in the SR model.

- **Generator Output Power (GOP)**

The stator winding temperature is related to the square of the phase current flowing in the windings. Therefore the higher the generated output power, the hotter the windings become. This independent variable is labelled as "Active Power" in the SR model.

- **Stator Winding Temperature (SWT)**

The stator winding temperature is the dependent variable, which the model regresses. Having knowledge which independent variable has the highest influence on stator winding temperature is important to optimise the generator operation. The SWT is predicted by the model based on the values of the independent variables AT, NT and GOP. The dependent variable is labelled as "Mean Winding Tmp U1" in the SR model.

6.4. Significance of regression model

The first check if the SR model is acceptable is to evaluate the value of the Coefficient of Determination R^2 ($0 \le R^2 \le 1$), which also means the goodness of fit test. It shows the proportion of the variation of the dependent variable explained by the independent variables. A value of R^2 close to 1 is ideal but it does not always imply that the model fits the data best or that future predictions by the model are perfect. It is affected by the number of independent variables, scatter or distribution of the independent variable(s) as well as adding higher polynomial values of the independent variable(s) in the model [24]. R^2 can be calculated using:

$$R^2 = \frac{SS_R}{SS_T} \tag{5}$$

where SS_R, regression sum of squares; SS_T, total sum of squares.

The F-test based on an F- distribution confirms the significance of the regression model. The following hypothesis is valid:

H_0: $\beta_1, \beta_2 \ldots . \beta_k = 0$.

H_1: $\beta_j \neq 0$.....for at least one j.

The F- critical value of the F- distribution is calculated in Microsoft Excel using the function:

$$\text{F.INV(probability, DoF 1, DoF 2)} \tag{6}$$

F.INV, calculates the inverse of the F-distribution; Probability, 95% confidence level; DoF 1, degrees of freedom. Number of independent variables; DoF 2, degrees of freedom. Number of residuals.

If F- critical >value needs to be larger than F—model value for the Null Hypothesis H_0 to be rejected which confirms that the model fits the data adequately with a 95% confidence level. Additionally the regression coefficients (β_0-β_3) in this model should all have p-value less than 0.05, which also confirms that H_0 can be rejected.

6.5. Analysis of residuals

The residuals also called the errors, are defined as the difference between the actual observation and the predicted observation from the model:

$$e_i = y_i - \hat{y}_i, i = 1, 2, \ldots\ldots n \tag{7}$$

where e_i, residual or error; y_i, actual observation; $\hat{y}_i$, - predicted observation from the model.

By plotting the residuals it can illustrate how the model best fit the data and show up any deviations from the previous assumptions made on applying linear regression. To check for normality in the residuals of the model, a normal probability plot of the residuals can be obtained in Microsoft Excel. A plot of the residuals versus the predicted observation $\hat{y}_i$ can also be retrieved in the same manner. This plot has to show the residuals outlined in a horizontal distribution about the zero average without any distinctive pattern for the model to be adequate [24]. Residual plots can have one of the four general outlines as shown in **Figure 5**. **Figure 5(a)** shows that the model is ideal, whereas the other plots (b–d) contain anomalies which show that the model could be inadequate for the data sample.

In this study a normal probability plot of residuals versus their standardised Z-scores is given. The procedure to construct the normal probability plot is as follows:

- Obtain the normal residuals from Microsoft Excel (SR);
- Rank each of the residuals;
- Calculate the percentile or proportion of the residuals that is smaller than a particular residual using:

$$P_i = \frac{Rankvalue - 0.375}{n + 0.25} \tag{8}$$

Where, n is the number of observations;

- Calculate the Z-score using Microsoft Excel Normal Distribution function:

$$= norm\ .\ s.\ inv(P_i) \tag{9}$$

- Print a scatter plot residuals vs. Z-scores.

This method is considered an improvement of the normal probability plot of the residuals in Microsoft Excel. If the residuals are normally distributed, 99.72% of the data will fall within 3 standard deviations of the mean. Therefore we can conclude that Z-score values outside these ranges do not have the same characteristics as the rest of the data and are possible outliers.

6.6. Intrinsically linear models

Linear regression can also be applied to investigate nonlinear characteristics between variables. Instead of using a straight line, linear regression has the functionality to fit curves to

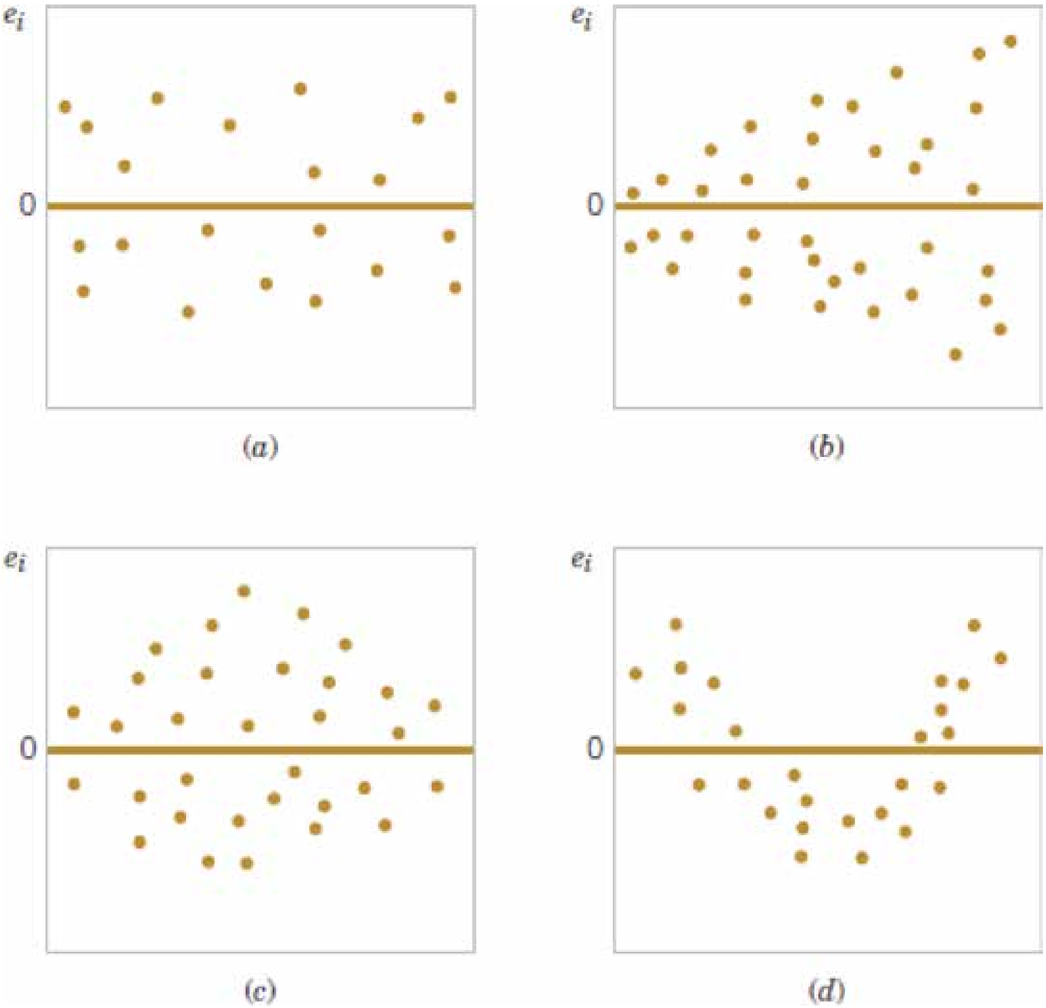


Figure 5. General residual plot patterns.

data which could be more appropriate for nonlinear conditions. In this case transformation of the dependent and or independent variables are required.

Intrinsically linear models or curve fitting the data can be done through polynomial regression where the independent variables are transformed in consecutive powers i.e. X, X^2, X^3 etc. Polynomial regression is used to detect any nonlinearity between the independent and dependent variables. Therefore the 2nd and 3rd powers of all three dependent variables AT, NT and GOP together with the linear values are used in the SR model.

A cubic polynomial with one independent variable has the following form:

$$Y = b_0 + b_1 X + b_2 X^2 + b_3 X^3 \tag{10}$$

If we set $x_1 = X$, $x_2 = X^2$, $x_3 = X^3$ then Eq. (8) can be rewritten as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \epsilon \tag{11}$$

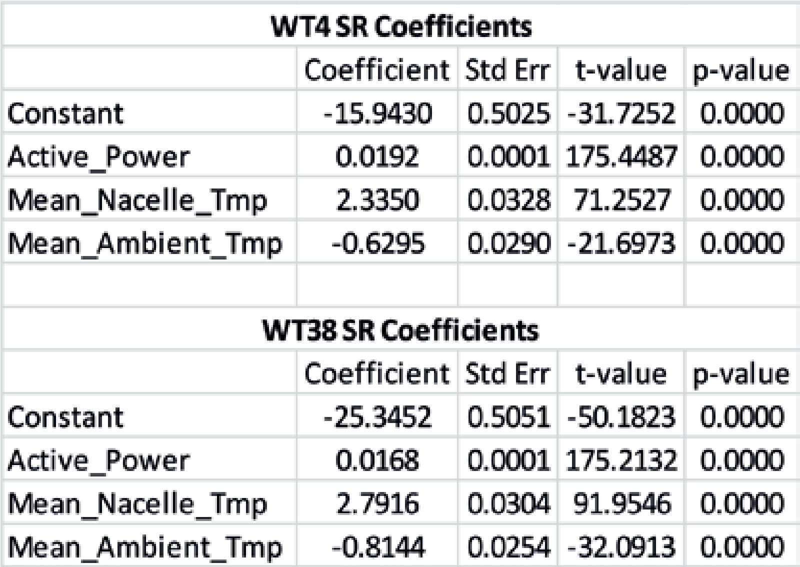
This is a multiple linear regression model similar to Eq. (2).

7. Statistical model analysis

Generator output power followed by nacelle temperature affects stator winding temperature the most as shown in **Table 3**.

This is expected as higher generated output power, cause more current flow through the windings and more heat is generated which is proportional to the square of the current. The nacelle temperature represents the ambient temperature of the generator and therefore also has a big impact. Insulation material of electrical machines is generally designed for an ambient temperature of 40°C and higher temperatures degrades the winding insulation material. Temperatures higher than 40°C in the nacelle can therefore cause the generator to shutdown to maintain the temperature raise limit of the insulation, which is 105°C for Class F. The temperature rise limit is calculated by subtracting the ambient temperature from the hot temperature of the insulation, which is 155–40°C. The outside temperature referred to as ambient temperature in the SR model is used for cooling of the stator windings. The outside air temperature has a limit of 45°C before the controller shuts down the machine to prevent overheating of the stator. Dirty or blocked air filters can also affect effective cooling. These can be checked during routine maintenance activities and replaced as required.

According to the SR model its ability to predict stator-winding temperature for WT4 and WT38 can be obtained using:



WT4 SR Coefficients				
	Coefficient	Std Err	t-value	p-value
Constant	-15.9430	0.5025	-31.7252	0.0000
Active_Power	0.0192	0.0001	175.4487	0.0000
Mean_Nacelle_Tmp	2.3350	0.0328	71.2527	0.0000
Mean_Ambient_Tmp	-0.6295	0.0290	-21.6973	0.0000
WT38 SR Coefficients				
	Coefficient	Std Err	t-value	p-value
Constant	-25.3452	0.5051	-50.1823	0.0000
Active_Power	0.0168	0.0001	175.2132	0.0000
Mean_Nacelle_Tmp	2.7916	0.0304	91.9546	0.0000
Mean_Ambient_Tmp	-0.8144	0.0254	-32.0913	0.0000

Table 3. Independent variables coefficients.

$$\mathrm{SWT} = 0.0192\mathrm{AP} + 2.335\mathrm{NT} - 0.6295\mathrm{AT} - 15.943\ (\mathrm{WT4}) \tag{12}$$

$$\mathrm{SWT} = 0.0168\mathrm{AP} + 2.7916\mathrm{NT} - 0.8144\mathrm{AT} - 25.3452\ (\mathrm{WT38}) \tag{13}$$

where AP, active power (Generator output power); NT, nacelle temperature; AT, ambient temperature.

It can be concluded that the location and wind resource of the two turbines have a significant impact on the stator winding temperature. Environmental conditions could be less ideal for one turbine, which effects the cooling of the nacelle and generator. Access to optimum wind conditions means a higher capacity factor and also higher average stator winding temperatures. The level of maintenance also needs consideration as one turbine can be exposed to severe dusty or moist conditions.

7.1. Adequacy of the SR model

Linear regression models such as SR need to meet certain criteria for accurate modelling of relationship between variables. It is generally assumed that these relationships between the variables are linear for the modelling to be successful.

7.1.1. Significance of the model

The coefficient of determination or R^2 indicates how well the independent variables explain the variability in the dependant variable. The SR model calculated $R^2 = 0.911$ for WT4 and $R^2 = 0.9234$ for WT38. Although the value of R^2 in both models is high, the ability of the models to predict stator-winding temperature accurately is not guaranteed. It does however indicate that GOP, NT and AT has a huge impact on the stator winding temperature.

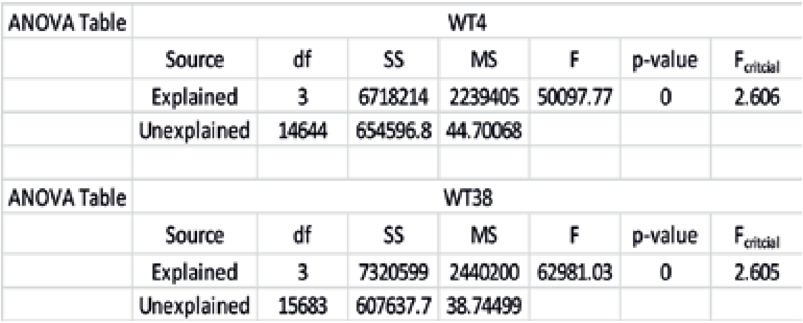
The F-test (value) confirms if the regression is significant. If the F-test falls to the left of the F-critical value in the F Distribution, the Null Hypothesis is accepted which means the regressors have no influence on the depended variable. If F-test > F-critical, the Null Hypothesis is rejected. The ANOVA Tables of both SR models in **Table 4** shows that the regression is significant which means the models for both wind turbines are adequate. In **Table 4** the p-values of the regressors are all less than 0.05, which also confirms the significance of the model.

7.1.2. Using intrinsically linear models

The use of intrinsically linear models allows linear regression to model nonlinear relationships through the transformation of the variables. In this study a 3rd degree polynomial regression model was applied to establish if it predicts stator-winding temperature more accurately than the straight-line model. The results of the polynomial regression models of WT4 and WT38 are shown in **Table 5**.

X—AT, NT and AP; X^2—ATT, NTT and APP; X^3—ATTT, NTTT and APPP.

The value of R^2 in the polynomial regression models show an improvement of less than 0.1% compared to the SR models. Therefore both models explain the variation in stator winding



ANOVA Table	WT4						
	Source	df	SS	MS	F	p-value	$F_{critcial}$
	Explained	3	6718214	2239405	50097.77	0	2.606
	Unexplained	14644	654596.8	44.70068			
ANOVA Table	WT38						
	Source	df	SS	MS	F	p-value	$F_{critcial}$
	Explained	3	7320599	2440200	62981.03	0	2.605
	Unexplained	15683	607637.7	38.74499			

Table 4. ANOVA statistics of SR.

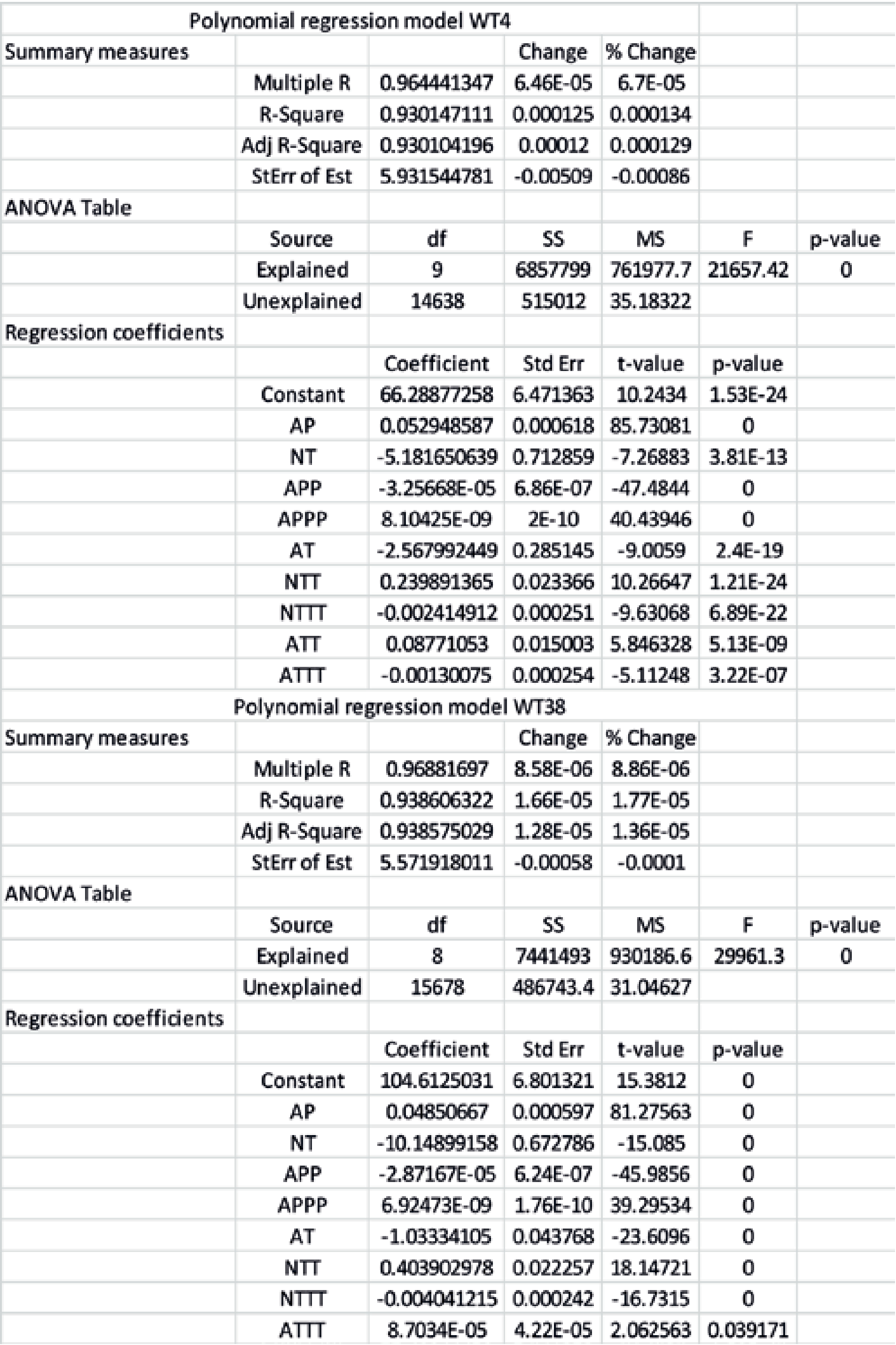
temperature by the independent variables with the same accuracy. The F-test of the SR model is much higher than the polynomial regression model, which means the SR model is more significant. The significance of the independent variables as determined by SR indicates that the linear independent variables are more important than the transformed independent variables. The SR model is simple, easy to implement and performs better than the polynomial regression model according to the various tests that were done. Considering the complexity and timeous development of the polynomial regression model, its application in this study is not justified.

7.1.3. Performance of the SR models

The regression model in this study is applied to identify abnormal high stator winding temperatures in the induction generator. Stator temperature SCADA logs of 10-minute intervals during November 2015 will be used as input to both wind turbine models. High stator winding temperatures outside the normal operating range of the generator can possibly be attributed to:

- Physical damage of the stator winding;
- Inadequate maintenance or cooling;
- Incorrect measurements,
- Equipment failure or
- Adverse operating conditions.

In WT4 where stator winding temperatures are below 40°C, the predicted temperatures by the SR model are higher than the actual temperatures. This over estimation can also be observed at the higher temperature ranges although the prediction errors are smaller. The SR model for WT38 has similar performances when the stator winding temperatures are below 40°C but has frequent under estimations at higher temperatures. The performances of the SR models for WT4 and WT38 are shown in **Figures 6** and **7**. Both models are able to predict the temperature trends in an acceptable manner and show very good accuracy when the stator winding temperatures are between 50°C and 90°C.



Polynomial regression model WT4						
Summary measures			Change	% Change		
	Multiple R	0.964441347	6.46E-05	6.7E-05		
	R-Square	0.930147111	0.000125	0.000134		
	Adj R-Square	0.930104196	0.00012	0.000129		
	StErr of Est	5.931544781	-0.00509	-0.00086		
ANOVA Table						
	Source	df	SS	MS	F	p-value
	Explained	9	6857799	761977.7	21657.42	0
	Unexplained	14638	515012	35.18322		
Regression coefficients						
		Coefficient	Std Err	t-value	p-value	
	Constant	66.28877258	6.471363	10.2434	1.53E-24	
	AP	0.052948587	0.000618	85.73081	0	
	NT	-5.181650639	0.712859	-7.26883	3.81E-13	
	APP	-3.25668E-05	6.86E-07	-47.4844	0	
	APPP	8.10425E-09	2E-10	40.43946	0	
	AT	-2.567992449	0.285145	-9.0059	2.4E-19	
	NTT	0.239891365	0.023366	10.26647	1.21E-24	
	NTTT	-0.002414912	0.000251	-9.63068	6.89E-22	
	ATT	0.08771053	0.015003	5.846328	5.13E-09	
	ATTT	-0.00130075	0.000254	-5.11248	3.22E-07	
Polynomial regression model WT38						
Summary measures			Change	% Change		
	Multiple R	0.96881697	8.58E-06	8.86E-06		
	R-Square	0.938606322	1.66E-05	1.77E-05		
	Adj R-Square	0.938575029	1.28E-05	1.36E-05		
	StErr of Est	5.571918011	-0.00058	-0.0001		
ANOVA Table						
	Source	df	SS	MS	F	p-value
	Explained	8	7441493	930186.6	29961.3	0
	Unexplained	15678	486743.4	31.04627		
Regression coefficients						
		Coefficient	Std Err	t-value	p-value	
	Constant	104.6125031	6.801321	15.3812	0	
	AP	0.04850667	0.000597	81.27563	0	
	NT	-10.14899158	0.672786	-15.085	0	
	APP	-2.87167E-05	6.24E-07	-45.9856	0	
	APPP	6.92473E-09	1.76E-10	39.29534	0	
	AT	-1.03334105	0.043768	-23.6096	0	
	NTT	0.403902978	0.022257	18.14721	0	
	NTTT	-0.004041215	0.000242	-16.7315	0	
	ATTT	8.7034E-05	4.22E-05	2.062563	0.039171	

Table 5. Polynomial regression model.

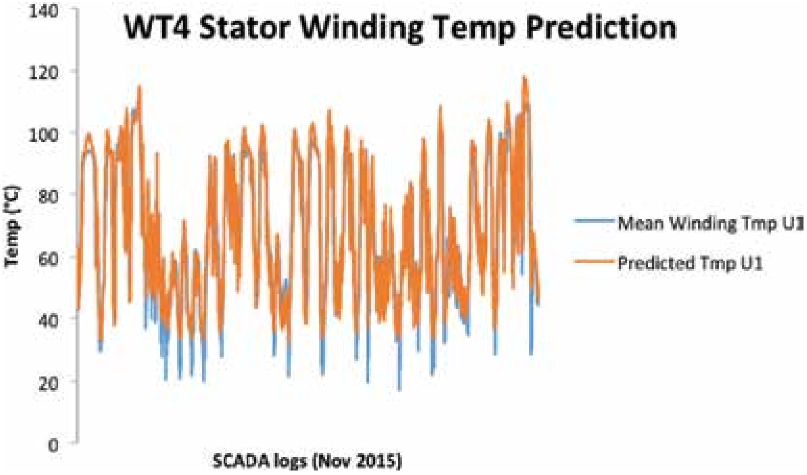


Figure 6. WT4 SR model performance.

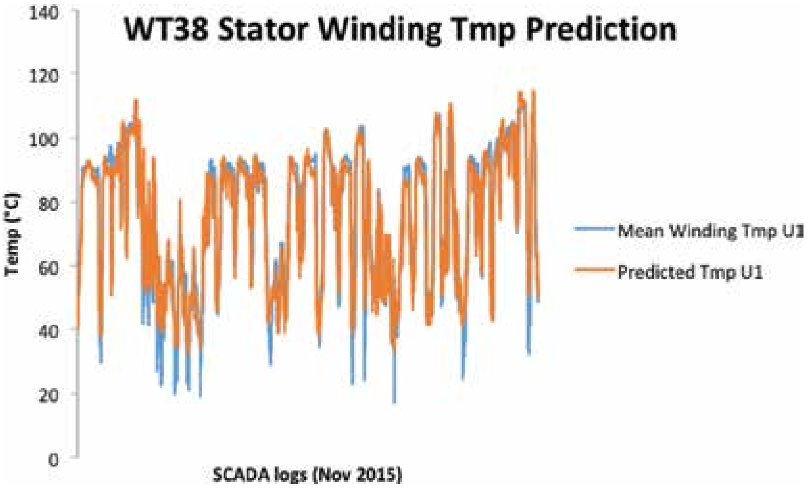


Figure 7. WT38 SR model performance.

The SR model deficiencies at the two extreme ends of the data distribution are possibly caused by nonlinear behaviour. These data points fall outside the three standard deviations of the normal distribution of temperature ranges as shown by **Figure 7**. There is a clear deviation by these data points away from the straight-line function used in SR model. Because wind turbines produces power below rated capacity the majority of the time, very low power regions just above the cut-in wind speed can result in different stator winding temperatures even if the environmental conditions are the same. These represent the stator winding temperatures below 40°C where the SR model performances are inadequate. Above rated speeds the wind turbine control system regulates its output power, which requires predominantly nonlinear control strategies. The rotor blade aerodynamics are changed rapidly to prevent excess power generation and loading on the wind turbine.

8. Conclusion

The aim of this study was to develop a new condition monitoring technique for stators in SCIGs. A statistical model was developed using SCADA data to estimate the relationships between winding temperatures and other variables. Predicting faults in stator windings are challenging because the unhealthy condition rapidly evolves into a functional failure. The analysis of SCADA data as a condition-monitoring tool for stator windings has been proven to be adequate. Active power, ambient and nacelle temperatures showed that the effects on stator winding temperature are significant as calculated by the statistical model. The capability of the model is proven in the analysis of the normal probability plots of the residuals, F-test and the value of R2. The statistical model performs very well when the wind turbine produces power at a constant rate below rated capacity. This operating region of the wind turbine has a more linear characteristic. Since a wind turbine spends the majority of the time in this operating region, the model can definitely be used as a conditioning monitoring tool for the SCIGs at Sere and similar wind farms.

Author details

Ian Kuiler, Marco Adonis* and Atanda Raji

*Address all correspondence to: adonism@cput.ac.za

Cape Peninsula University of Technology, Cape Town, South Africa

References

[1] IRENA. Renewable Energy Benefits: Leveraging Local Capacity for Onshore Wind. Abu Dhabi: International Renewable Energy Agency; 2017

[2] IRENA. Renewable Power Generation Costs in 2017. Abu Dhabi: International Renewable Energy Agency; 2018

[3] Wu B, Lang Y, Zargari N, Kouro S. Power Conversion and Control of Wind Energy Systems. Piscataway, NJ, USA: IEEE Press; John Wiley & Sons; 2011

[4] Boccard N. Capacity factor of wind power realized values vs. estimates. Energy Policy. 2009;**37**(7):2679-2688

[5] EWEA. The economics of wind energy. Renewable and Sustainable Energy Reviews. 2009

[6] Lazard, Lazard's Levelized Cost of Energy Analysis. Version 11.0; November 2017

[7] Saheb-Koussa D, Haddadi M, Belhamel M, Koussa M, Noureddine S. Modeling and simulation of wind generator with fixed speed wind turbine under MATLAB-Simulink. Energy Procedia. 2012;**18**:701-708

[8] Bustos G, Vargas LS, Milla F, Sáez D, Zareipour H, Nuñez A. Comparison of fixed speed wind turbines models: A case study. In: Proceedings of the 38th Annual Conference of

the IEEE Industrial Electronics Society (IECON-2012), Montreal, Canada, October 25-28. 2012

[9] Murthy SS, Singh B, Goel PK, Tiwari SK. A comparative study of fixed speed and variable speed wind energy conversion systems feeding the grid. In: 7th International Conference on Power Electronics and Drive Systems. 2007. pp. 736-743

[10] Pao LY, Johnson KE. A tutorial on the dynamics and control of wind turbines and wind farms. In: American Control Conference, 2009. ACC'09. 2009. pp. 2076-2089

[11] Serrano-Gonzalez J, Lacal-Arantegui R. Technological evolution of onshore wind turbines—A market-based analysis. Wind Energy. 2016, 2016;**19**(12):2171-2187

[12] Mohammad S, Das N, Roy S. A review of the state of the art of generators and power electronics converter topologies for wind. 2013;**3**(3):283-291

[13] Soter S, Wegener R. Development of induction machines in wind power technology. IEEE International Electric Machines & Drives Conference. 2007;**2**:1490-1495

[14] Chen Y, Pillay P, Khan A, Member S. PM wind generator topologies. IEEE Transactions on Industry Applications. 2005;**41**(6):1619-1626

[15] Gowdar RD, Mallikarjune Gowda MC. Reasons for wind turbine generator failures: A multi-criteria approach for sustainable power production. Renewables: Wind, Water, and Solar. 2016;**3**(1):9

[16] Hahn B, Durstewitz M, Rohrig K. Reliability of wind turbines. In: Wind Energy. 2007. Berlin Heidelberg: Springer; pp. 329-332. DOI: 10.1007/978-3-540-33866-6_62

[17] Shipurkar U, Ma K, Polinder H, Blaabjerg F, Ferreira JA. A review of failure mechanisms in wind turbine generator systems. In: 17th European Conference on Power Electronics and Applications (EPE'15 ECCE-Europe), Geneva. 2015. pp. 1-10

[18] Alewine K, Chen W. A review of electrical winding failures in wind turbine generators. IEEE Electrical Insulation Magazine. 2012;**28**(4):8-13

[19] Gao G, Chen W. Design challenges of wind turbine generators. In: IEEE Electr. Insul. Conf., no. June. 2009. pp. 146-152

[20] Tian Z, Jin T, Wu B, Ding F. Condition based maintenance optimization for wind power generation systems under continuous monitoring. Renewable Energy. 2011;**36**(5): 1502-1509

[21] Agrawal KK, Pandey GN, Chandrasekaran K. Analysis of the condition based monitoring system for heavy industrial machineries. In: IEEE International Conference on Computational Intelligence and Computing Research, Enathi. 2013. pp. 1-4

[22] Yang W, Tavner PJ, Crabtree CJ, Feng Y, Qiu Y. Wind turbine condition monitoring: Technical and commercial challenges. Wind Energy. 2014;**17**(5):673-693

[23] Amirat Y, Benbouzid MEH, Al-Ahmar E, Bensaker B, Turri S. A brief status on condition monitoring and fault diagnosis in wind energy conversion systems. Renewable and Sustainable Energy Reviews. 2009;**13**(9):2629-2636

[24] Montgomery D, Runger G. Applied Statistics and Probability for Engineers. 6th ed. Hoboken, NJ, USA: Wiley; 2014

Chapter 3

Hybrid Fault Diagnosis Method Based on Mechanical-Electrical Intersectional Characteristics for Generators

Yu-Ling He and Yue-Xin Sun

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.79955

Abstract

In this chapter, a new hybrid fault diagnosis method based on the mechanical-electrical intersectional characteristics for turbo-generators is proposed. Different from other studies, this method not only employs the rotor vibration characteristics but also uses the stator vibration features and the circulating current properties inside the parallel branches of the same phase. Detailed theoretical analysis, as well as the experimental verification study, is carried out to demonstrate the proposed method. It is shown that in the proposed criterion for the method, the combining faulty characteristics for the single rotor eccentricity fault, the single rotor interturn short circuit fault, and the composite fault composed of the rotor eccentricity and the rotor interturn short circuit are all unique. The running conditions can be accurately and quickly identified by the proposed method. The work proposed in this chapter offers a new thought for the condition monitoring and the fault diagnosis of generators.

Keywords: turbo-generator, rotor eccentricity, rotor interturn short circuit, mechanical-electrical intersectional characteristics

1. Introduction

The generator is the key equipment for a power plant and needs timely and accurate monitoring and maintaining. Typically, after a long-term operation, generators may endure many electrical faults such as the rotor interturn short circuit (RISC) [1] and the stator interturn short circuit [2, 3], as well as mechanical faults such as rotor eccentricity [4].

RISC is the fault in which short circuit takes place between two adjacent turns inside the filed winding in the rotor [5]. That means, only the insulation between the two neighboring turns is damaged, while the main insulation of the whole winding bar is still fine.

Rotor eccentricity is the fault that occurs when the air-gap between the rotor and the stator is not average [6]. This fault can be further divided into three types, i.e., the static rotor eccentricity in which the minimum air-gap remains stable in a certain direction, the dynamic rotor eccentricity in which the minimum air-gap will vary as the rotor rotates, and the mixed rotor eccentricity in which static and the dynamic eccentricity occur at the same time. Generally, dynamic eccentricity is more complex and usually has a small eccentricity value, while static eccentricity is more common and more likely to have a larger eccentricity value caused by many factors such as bearing damage or un-accordance, assembling error, and deformation of the stator core. In this chapter, we mainly focus on the static rotor eccentricity.

So far, scholars have developed many monitoring and diagnosis methods for either the rotor eccentricity fault or the RISC fault. As for the rotor eccentricity fault, studies on the monitoring and diagnosis of air-gap eccentricity primarily focus on the stator current [7] and voltage [8, 9], the rotor current [9] and the shaft voltage [10], the inductance variation of the windings [11, 12], and the rotor UMP and vibration analysis [13–15]. The inductance variation analysis is mainly based on the winding function theory [16] and the improved winding function theory [17, 18] and needs a large amount of calculation, while the current and the voltage analysis is actually based on the harmonic changes of the magnetic flux density [7]. Using a direct analysis of the spectrum of the stator and rotor current or voltage obtained via Fourier transform, it is sometimes hard to exactly identify the eccentricity due to the inconspicuous amplitude changes compared with the noise signal magnitude, especially when the capacity of the generator is small or the eccentricity is not so severe. To overcome this disadvantage, scholars have developed an improved method using search coils [19].

People have studied the theoretical deduction and the simulation analysis of RISC in wind-powered generator [20, 21] and analyzed the change rate of the magnetic flux as well in order to detect this fault [22]. It is found that the induced voltage in rotor can be used to predict the location and the number of short circuit turns [23]. Meanwhile, researchers have also studied the characteristics of the excitation currents [24, 25], the copper losses [24], and the unbalanced magnetic pull (UMP) for the interturn short circuit monitoring [15, 26, 27]. Generally, at present, the application of search coils, which is mainly based on the magnetic field density (MFD) variation, is still adopted as a primary approach to monitor and diagnose this fault [28–30]. Therefore, further investigation on MFD variation characteristics at great length is of significance and will be the key to improve the monitoring level of the very failure. It is shown that some specific harmonic characteristics are very helpful and even more effective than other traditional means to diagnose the fault [31, 32].

However, since the actual performing condition is far more complex than the ideal normal condition and the single fault cases, the generator may display some untypical fault characteristics. For example, in addition to the RISC fault, the generator may have rotor eccentricity as well, i.e., the composite fault composed by rotor eccentricity and RISC. In this case, the fault characteristics are not the same as those of the single RISC fault or the single rotor eccentricity fault. Thus the problem is to identify and diagnose the very fault (the single rotor eccentricity, the single RISC, and the composite fault) accurately.

In this chapter, we will discuss a new method combining the mechanical fault characteristics, i.e., the stator and the rotor vibration characteristics, with the electrical fault characteristics, i.e., the circulating current inside the parallel branches of the same phase (CCPB), to identify the single rotor eccentricity fault, the single RISC fault, and the composite fault composed by these two.

2. Theoretical analysis

As is widely comprehended, the electrical and the mechanical properties of the generator are all closely related to the magnetic flux density (MFD). For example, the magnetic pulls acting on the stator core and the rotor core are in proportion to the square of MFD, and the stator current and voltage are in direct proportion to MFD. Generally, both the rotor eccentricity fault and the RISC fault will affect MFD.

In this section, we will firstly analyze the impact of the four running conditions, i.e., the normal condition, the single rotor eccentricity fault, the single RISC fault, and the composite fault composed of rotor eccentricity and RISC. Then, the unbalanced magnetic pull (UMP) formulas and the electromotive force difference expression between the two parallel branches will be deduced in detail to obtain the qualitative theoretical results.

2.1. MFD study for each case

MFD is composed by two factors, the magnetomotive force (MMF) and the permeance per unit area. Usually MFD is written as

$$B(\alpha_m, t) = f(\alpha_m, t)\Lambda(\alpha_m, t) \tag{1}$$

where $f(\alpha_m, t)$ is the MMF, and $\Lambda(\alpha_m, t)$ is the permeance per unit area.

Typically, RISC primarily affects MMF but has little impact on the permeance, while the rotor eccentricity mainly impacts on the permeance per unit area but has little influence on MMF. In normal condition, there is neither RISC nor rotor eccentricity. In this case, the air-gap can be indicated as **Figure 1(a)**, while the rotor MMF and the vector diagram of the stator and rotor MMFs can be indicated as given in **Figure 2(a)** and **Figure 3(a)**, respectively, by the shorted turns.

As indicated in **Figure 3(a)**, the MMF in normal condition can be written as

$$\begin{cases} f(\alpha_m, t) = F_r \cos(\omega t - \alpha_m) + F_s \cos\left(\omega t - \alpha_m - \frac{\pi}{2} - \psi\right) = F_1 \cos\left(\omega t - \alpha_m - \beta\right) \\ F_1 = \sqrt{(F_r - F_s \sin\psi)^2 + (F_s \cos\psi)^2} \\ \beta = \arctan\dfrac{F_s \cos\psi}{F_r - F_s \sin\psi} \end{cases} \tag{2}$$

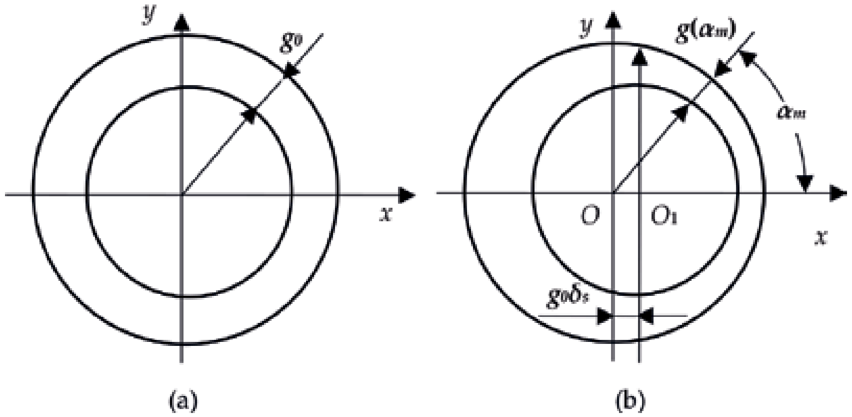


Figure 1. Air-gap for the four performing conditions. (a) Normal condition and RISC and (b) rotor eccentricity and composite fault.

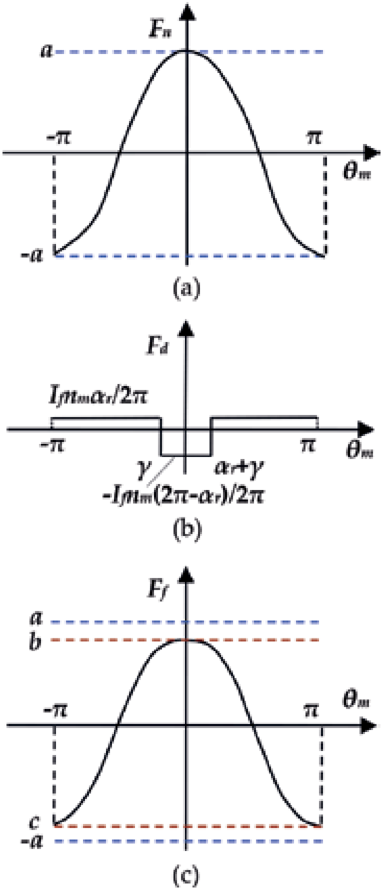


Figure 2. Rotor MMF before and after RISC. (a) Normal rotor MMF, (b) inverse MMF produced, and (c) MMF under RISC.

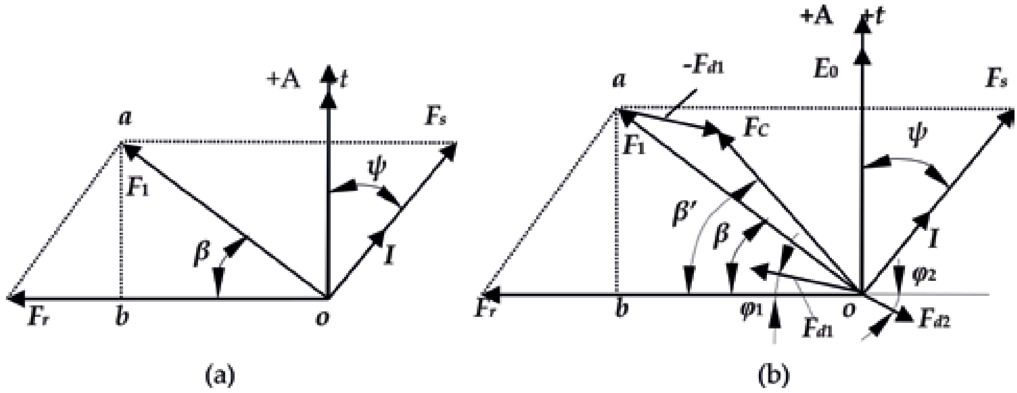


Figure 3. Vector diagram of the stator and rotor MMFs for the four performing conditions. (a) Normal condition and rotor eccentricity and (b) RISC and composite fault.

Since the permeance per unit area is in inverse proportion to the radial air-gap length, the permeance per unit area in normal condition can be written as

$$\Lambda(\alpha_m, t) = \frac{\mu_0}{g(\alpha_m, t)} = \frac{\mu_0}{g_0} = \Lambda_0 \tag{3}$$

In the case of rotor eccentricity, the MMF is the same as in normal condition, while the permeance per unit area is

$$\begin{cases} \Lambda(\alpha_m, t) = \dfrac{\mu_0}{g(\alpha_m, t)} = \dfrac{\mu_0}{g_0(1-\delta_s \cos\alpha_m)} = \dfrac{\mu_0}{g_0}\left(1+\delta_s\cos\alpha_m+\delta_s^2\cos^2\alpha_m+\cdots\right) \\ \qquad \approx \Lambda_0(1+\delta_s\cos\alpha_m) = \Lambda_0 + \Lambda_s\cos\alpha_m \\ \Lambda_s = \Lambda_0\delta_s \end{cases} \tag{4}$$

where μ_0 is the permeability of the air, g_0 is the radial air-gap length, and δ_s is the relative rotor eccentricity.

In the case of RISC, since there is no longer exciting current in the shorted turns, it is equivalent to adding an inverse current to the normal exciting current for the shorted turns. The rotor MMF before and after RISC is indicated in **Figure 2**. Since the area of the produced positive MMF should be equal to that of the induced negative MMF, the inverse MMFs produced by the shorted turns can be written as

$$f_d(\theta_m) = \begin{cases} -\dfrac{I_f n_m(2\pi-\alpha_r)}{2\pi} & \gamma \le \theta_m \le \gamma+\alpha_r \\ \dfrac{I_f n_m \alpha_r}{2\pi} & other\ condition \end{cases} \tag{5}$$

where θ_m is the circumferential angle on the rotor surface, I_f is the exciting current of the generator, n_m is the number of the shorted turns, γ is the circumferential angle to indicate the beginning RISC position, and α_r is the angle between the two slots where RISC takes place.

$F_d(\theta_m)$ can be expanded by Fourier series and written as

$$\begin{cases} f_d(\theta_m) = A_0 + \sum\limits_{n=1}^{\infty}[A_n \cos(n\theta_m) + B_n \sin(n\theta_m)] \\ A_0 = \dfrac{1}{2\pi}\displaystyle\int_0^{2\pi} F_d(\theta_m)d\theta_m = 0 \\ A_n = \dfrac{1}{\pi}\displaystyle\int_0^{2\pi} F_d(\theta_m)\cos(n\theta_m)d\theta_m = -\dfrac{I_f n_m[\sin(n(\alpha_r + \gamma)) - \sin(n\gamma)]}{n\pi} \\ B_n = \dfrac{1}{\pi}\displaystyle\int_0^{2\pi} F_d(\theta_m)\sin(n\theta_m)d\theta_m = \dfrac{I_f n_m[\cos(n(\alpha_r + \gamma)) - \cos(n\gamma)]}{n\pi} \end{cases} \tag{6}$$

Since the nth MMF harmonic is equivalent to the main MMF which is produced by the generator that has n pole-pairs, the reverse MMF induced by the short circuit turns can be written as a function which is both time and space dependent.

$$\begin{cases} F_d(\alpha_m, t) = \sum\limits_{n=1}^{\infty}[A_n \cos n(\omega t - \alpha_m) + B_n \sin n(\omega t - \alpha_m)] = \sum\limits_{n=1}^{\infty} F_{dn} \cos n(\omega t - \alpha_m - \varphi_n) \\ F_{dn} = \sqrt{{A_n}^2 + {B_n}^2} \\ \varphi_n = \arctan\dfrac{B_n}{A_n} \end{cases} \tag{7}$$

Correspondingly, ignoring the higher-order harmonics, according to **Figure 3(b)**, the MMF under RISC can be written as

$$\begin{cases} f(\alpha_m, t) = F_1 \cos(\omega t - \alpha_m - \beta) - F_{d1} \cos(\omega t - \alpha_m - \varphi_1) - F_{d2} \cos(\omega t - \alpha_m - \pi - \varphi_2) \\ \qquad\qquad = F_C \cos\cos(\omega t - \alpha_m - \beta') - F_{d2} \cos(\omega t - \alpha_m - \pi - \varphi_2) \\ F_C = \sqrt{(F_r - F_s \sin\psi - F_{d1}\cos\varphi_1)^2 + (F_s \cos\psi - F_{d1} \sin\varphi_1)^2} \\ \beta' = \arctan\dfrac{F_s \cos\psi - F_{d1}\sin\varphi_1}{F_r - F_s \sin\psi - F_{d1}\cos\varphi_1} \end{cases} \tag{8}$$

Feeding Eqs. (2)–(4) and (8) into (1), the MFDs for the four running conditions can be obtained.

$$B(\alpha_m, t) = \begin{cases} F_1 \cos(\omega t - \alpha_m - \beta)\Lambda_0 \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \text{normal} \\ F_1 \cos(\omega t - \alpha_m - \beta)(\Lambda_0 + \Lambda_s \cos\alpha_m) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \text{rotor eccentricity} \\ [F_C \cos\cos(\omega t - \alpha_m - \beta') - F_{d2}\cos(\omega t - \alpha_m - \pi - \varphi_2)]\Lambda_0 \cdot \cdots\cdots\cdots \text{RISC} \\ [F_C \cos\cos(\omega t - \alpha_m - \beta') - F_{d2}\cos(\omega t - \alpha_m - \pi - \varphi_2)] \\ \quad \times (\Lambda_0 + \Lambda_s \cos\alpha_m) \qquad\qquad\qquad\qquad \cdot \cdots\cdots\cdots \text{composite fault} \end{cases} \tag{9}$$

2.2. Mechanical-electrical characteristic analysis

The physical model of the stator core is a hollow shell, as indicated in **Figure 4**, while the physical model of the rotor core is the rigid cylinder. Therefore, the essential exciting force for the stator core is the magnetic pull per unit area (MPPUA), while for the rotor it is the integral

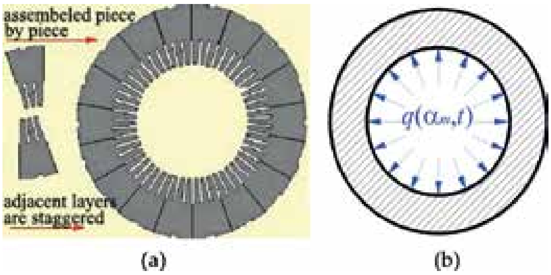


Figure 4. Structure and magnetic force of stator core. (a) Structure of stator core and (b) MPPUA on stator core.

force. This may not be easy to understand, but it can be suggested from **Figure 4(b)** that though the integral force of the stator core is zero due to the symmetric distribution of the unit force, it will still have periodical shrinking-expanding deformations, i.e., the radial vibrations, due to the cyclical pulsating feature of the MPPUA. However, for the rotor core, the MPPUA is not enough to cause radial vibrations for this solid cylinder. Therefore, the internal force, i.e., the unbalanced magnetic pull (UMP) will be the essential exciting force for the rotor.

The MPPUA and the UMP can be calculated via

$$\begin{cases} q(\alpha_m,t) = \dfrac{[B(\alpha_m,t)]^2}{2\mu_0} & \text{MPPUA} \\ \begin{cases} F_X = LR\displaystyle\int_0^{2\pi} q(\alpha_m,t)\cos\alpha_m d\alpha_m \\ F_Y = LR\displaystyle\int_0^{2\pi} q(\alpha_m,t)\sin\alpha_m d\alpha_m \end{cases} & \text{UMP} \end{cases} \tag{10}$$

Feeding Eq. (9) into (10), the exciting force needed to cause stator vibration and rotor vibration can be, respectively, written as Eqs. (11) and (12).

Since the stator and rotor vibration will have the same frequency harmonic components as the exciting force, as indicated in Eq. (11), the stator will have second harmonic vibrations in normal condition and in the rotor eccentricity case, while it will have first, second, third, and fourth harmonic components under RISC and the composite fault. Obviously, it is hard to identify the fault type accurately only by means of the stator vibration.

Comparing MPPUA formulas in the four running conditions in Eq. (11), it can be found that the magnitude of the second harmonic under rotor eccentricity fault will be larger than that of the normal condition because extra second harmonic terms are added in the formula. However, the magnitude of the second harmonic MPPUA under RISC will be smaller than that in normal condition because F_C is smaller than F_1, see **Figure 3(b)**. For the composite fault, the second harmonic MPPUA magnitude will be smaller than that under rotor eccentricity but larger than that in the RISC case. Since there is theoretically no fourth harmonic in normal condition when only considering the first MMFs, the occurrence of RISC will increase the fourth harmonic MPPUA.

The rotor, as indicated in Eq. (12), will have no vibrations in normal condition, while it will endure second harmonic vibrations in the case of rotor eccentricity; first harmonic vibration under RISC; and first, second, and third harmonic vibrations under the composite fault. It seems that the four running conditions can be identified by the rotor vibration characteristics. However, still other faults have the same rotor vibration features, leading to practical difficulties for exact diagnosis. For example, the mass imbalance fault will also cause the rotor to vibrate at the fundamental frequency, and the shaft misalignment fault will also lead to the rotor's second harmonic vibrations. Therefore, it is actually still hard to diagnose the fault accurately by only using the rotor vibration properties.

In fact, in addition to the stator and rotor vibrations, the circulating current inside the parallel branches (CCPB) of the same phase will vary as well due to different running conditions. Taking the SDF-9 type generator which will be employed as the study object behind as an example, the parallel branches and the CCPB in Phase A are indicated in **Figure 5**.

$$q(\alpha_m, t) = \left\{ \begin{array}{l}
\dfrac{{F_1}^2}{8\mu_0}\left[2\Lambda_0^2 + 2\Lambda_0^2 \cos\left(2\omega t - 2\alpha_m - 2\beta\right)\right] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{normal condition} \\
\dfrac{{F_1}^2}{8\mu_0}\Bigg\{ \left[\left(2\Lambda_0^2 + \Lambda_s^2\right) + \left(4\Lambda_0\Lambda_s \cos\alpha_m\right) + \left(\Lambda_s^2 \cos 2\alpha_m\right)\right] \\
\qquad + [0.5\Lambda_s^2 \cos\left(2\omega t - 2\beta\right) + 2\Lambda_0\Lambda_s \cos\left(2\omega t - \alpha_m - 2\beta\right) \\
\qquad + \left(2\Lambda_0^2 + \Lambda_s^2\right)\cos\left(2\omega t - 2\alpha_m - 2\beta\right) + 2\Lambda_0\Lambda_s \cos\left(2\omega t - 3\alpha_m - 2\beta\right) \\
\qquad + 0.5{\Lambda_s}^2 \cos\left(2\omega t - 4\alpha_m - 2\beta\right)]\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{rotor eccentricity} \\
\dfrac{\Lambda_0^2}{4\mu_0}[F_C^2 + F_{d2}^2 - 2F_C F_{d2} \cos\left(\omega t - \alpha_m + \beta_1 - 2\varphi_2\right) \\
\qquad + F_C^2 \cos 2\left(\omega t - \alpha_m - \beta\right) - 2F_C F_{d2} \cos\left(3\omega t - 3\alpha_m - \beta_1 - 2\varphi_2\right) \\
\qquad + F_{d2}^2 \cos 4\left(\omega t - \alpha_m - \varphi_2\right)] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{RISC} \\
\dfrac{1}{8\mu_0}\Bigg\{ \left[\left(F_C^2 + F_{d2}^2\right)\left(2\Lambda_0^2 + \Lambda_s^2\right) + 4\left(F_C^2 + F_{d2}^2\right)\Lambda_0\Lambda_s \cos\alpha_m + \left(F_C^2 + F_{d2}^2\right)\Lambda_s^2 \cos 2\alpha_m\right] \\
\qquad + [-4F_C F_{d2}\Lambda_0\Lambda_s \cos\left(\omega t + \beta_1 - 2\varphi_2\right) - 2F_C F_{d2}\left(2\Lambda_0^2 + \Lambda_s^2\right)\cos\left(\omega t - \alpha_m + \beta_1 - 2\varphi_2\right) \\
\qquad - F_C F_{d2}\Lambda_s^2 \cos\left(\omega t + \alpha_m + \beta_1 - 2\varphi_2\right) - 4F_C F_{d2}\Lambda_0\Lambda_s \cos\left(\omega t - 2\alpha_m + \beta_1 - 2\varphi_2\right) \\
\qquad - F_C F_{d2}\Lambda_s^2 \cos\left(\omega t - 3\alpha_m + \beta_1 - 2\varphi_2\right)] + [0.5F_C^2\Lambda_s^2 \cos\left(2\omega t - 2\beta_1\right) \\
\qquad + 2F_C^2\Lambda_0\Lambda_s \cos\left(2\omega t - \alpha_m - 2\beta_1\right) + F_C^2\left(2\Lambda_0^2 + \Lambda_s^2\right)\cos\left(2\omega t - 2\alpha_m - 2\beta_1\right) \\
\qquad + 2F_C^2\Lambda_0\Lambda_s \cos\left(2\omega t - 3\alpha_m - 2\beta_1\right) + 0.5F_C^2\Lambda_s^2 \cos\left(2\omega t - 4\alpha_m - 2\beta_1\right)] \\
\qquad + [-F_C F_{d2}\Lambda_s^2 \cos\left(3\omega t - \alpha_m - \beta_1 - 2\varphi_2\right) - 4F_C F_{d2}\Lambda_0\Lambda_s \cos\left(3\omega t - 2\alpha_m - \beta_1 - 2\varphi_2\right) \\
\qquad - 2F_C F_{d2}\left(2\Lambda_0^2 + \Lambda_s^2\right)\cos\left(3\omega t - 3\alpha_m - \beta_1 - 2\varphi_2\right) \\
\qquad - 4F_C F_{d2}\Lambda_0\Lambda_s \cos\left(3\omega t - 4\alpha_m - \beta_1 - 2\varphi_2\right) - F_C F_{d2}\Lambda_s^2 \cos\left(3\omega t - 5\alpha_m - \beta_1 - 2\varphi_2\right)] \\
\qquad + [0.5F_{d2}^2\Lambda_s^2 \cos\left(4\omega t - 2\alpha_m - 4\varphi_2\right) + 2F_{d2}^2\Lambda_0\Lambda_s \cos\left(4\omega t - 3\alpha_m - 4\varphi_2\right) \\
\qquad + F_{d2}^2\left(2\Lambda_0^2 + \Lambda_s^2\right)\cos\left(4\omega t - 4\alpha_m - 4\varphi_2\right) + 2F_{d2}^2\Lambda_0\Lambda_s \cos\left(4\omega t - 5\alpha_m - 4\varphi_2\right) \\
\qquad + 0.5F_{d2}^2\Lambda_s^2 \cos\left(4\omega t - 6\alpha_m - 4\varphi_2\right)]\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{composite fault}
\end{array} \right. \tag{11}$$

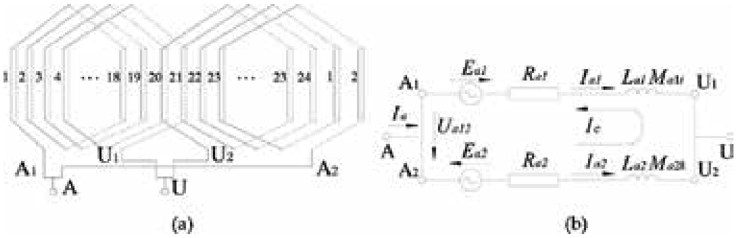


Figure 5. Parallel branch and CCPB in Phase A. (a) Winding distribution of Phase A and (b) equal circuit of parallel branches of Phase A.

The electromotive force difference between the two branches, which is indicated in **Figure 5(b)** and is the exciting source of the CCPB, can be obtained via Eq. (13), where E_{a1} and E_{a2} are the electromotive forces of the two branches, respectively; q is the number of slots for each pole per phase; w_c is the number of turns for each branch winding; k_{w1} is the fundamental frequency winding factor; τ is the polar distance; l is the effective length of the winding; and f is the electrical frequency. Feeding Eq. (9) into (13), the electromotive force difference can be obtained and written as in Eq. (14).

As indicated in Eq. (14), the CCPB has different features due to varied running conditions. However, it has the similar problem as the rotor vibration while using it as the fault diagnosis criterion. For example, due to the initial asymmetry inside the generator, the generator may have first harmonic CCPB in normal condition. Consequently, the CCPB features will be very similar not only between normal condition and rotor eccentricity, but also between RISC and the composite fault. Thus, it is still not enough to only use CCPB difference to identify the faults accurately.

$$\left\{\begin{array}{l}
\left.\begin{array}{l}
F_X = LR\int_0^{2\pi} q(\alpha_m,t)\cos\alpha_m d\alpha_m = 0 \\
F_Y = LR\int_0^{2\pi} q(\alpha_m,t)\sin\alpha_m d\alpha_m = 0
\end{array}\right\}\ldots\ldots\ldots\ldots\ldots\ldots\ldots\text{normal condition} \\
\left.\begin{array}{l}
F_X = \dfrac{LRF_1^2\pi}{4\mu_0}\left[2\Lambda_0\Lambda_s + 2\Lambda_0\Lambda_d\cos\omega t + \Lambda_0\Lambda_d\cos\left(\omega t - 2\beta\right)\right. \\
\qquad\qquad \left.+\Lambda_0\Lambda_s\cos\left(2\omega t - 2\beta\right)\right] \\
F_Y = \dfrac{LRF_1^2\pi}{4\mu_0}\left[2\Lambda_0\Lambda_d\sin\omega t + \Lambda_0\Lambda_d\sin\left(\omega t - 2\beta\right)\right. \\
\qquad\qquad \left.+\Lambda_0\Lambda_s\sin\left(2\omega t - 2\beta\right)\right]
\end{array}\right\}\ldots\ldots.\text{rotor eccentricity} \\
\left.\begin{array}{l}
F_X = LR\int_0^{2\pi} q(\alpha_m,t)\cos\alpha_m d\alpha_m = \dfrac{-F_CF_{d2}LR\Lambda_0^2\pi}{2\mu_0}\cos\left(\omega t + \beta_1 - 2\varphi_2\right) \\
F_Y = LR\int_0^{2\pi} q(\alpha_m,t)\sin\alpha_m d\alpha_m = \dfrac{-F_CF_{d2}LR\Lambda_0^2\pi}{2\mu_0}\sin\left(\omega t + \beta_1 - 2\varphi_2\right)
\end{array}\right\}\ldots\ldots\ldots\text{RISC} \\
\left.\begin{array}{l}
F_X = \dfrac{LR\pi}{8\mu_0}\left\{\left[4\left({F_C}^2 + {F_{d2}}^2\right)\Lambda_0\Lambda_s\right]\right. \\
\qquad +\left[-2F_CF_{d2}\left(2\Lambda_0^2 + 1.5\Lambda_s^2\right)\cos\left(\omega t + \beta_1 - 2\varphi_2\right)\right] \\
\qquad \left.+\left[2F_C^2\Lambda_0\Lambda_s\cos\left(2\omega t - 2\beta_1\right)\right] - \left[F_CF_{d2}\Lambda_s^2\cos\left(3\omega t - \beta_1 - 2\varphi_2\right)\right]\right\} \\
F_Y = \dfrac{LR\pi}{8\mu_0}\left\{\left[-2F_CF_{d2}\left(2\Lambda_0^2 + 0.5\Lambda_s^2\right)\sin\left(\omega t + \beta_1 - 2\varphi_2\right)\right]\right. \\
\qquad \left.+\left[2F_C^2\Lambda_0\Lambda_s\sin\left(2\omega t - 2\beta_1\right)\right] - \left[F_CF_{d2}\Lambda_s^2\sin\left(3\omega t - \beta_1 - 2\varphi_2\right)\right]\right\}
\end{array}\right\}..\text{compostie fault}
\end{array}\right. \tag{12}$$

$$\begin{cases} U_{a12}(\alpha_m,t) = -E_{a1}(\alpha_m,t) + j\omega L_{a1}I_{a1} + R_{a1}I_{a1} + j\omega\sum\limits_i M_{a1i}I_i \\ \qquad\qquad -j\omega\sum\limits_k M_{a2k}I_k - R_{a2}I_{a2} - j\omega L_{a2}I_{a2} - E_{a2}(\alpha_m,t) \\ E_{a1}(\alpha_m,t) = 2qw_c k_{w1}\tau l f B(\alpha_m,t) \\ E_{a2}(\alpha_m,t) = 2qw_c k_{w1}\tau l f B[(\alpha_m-\pi),t] \end{cases} \tag{13}$$

$$U_{a12} = \begin{cases} 0\cdot\cdots\cdots\cdots\cdots\text{normal condition} \\ -4qw_c k_{w1}\tau l f F_1\Lambda_s\cos\alpha_m\cos\left(\omega t-\alpha_m-\beta\right)\cdot\cdots\cdots\text{rotor eccentricity} \\ 4qw_c k_{w1}\tau l f F_{d2}\Lambda_0\cos 2\left(\omega t-\alpha_m-\varphi_2\right)\cdot\cdots\cdots\cdots\text{RISC} \\ 2qw_c k_{w1}\tau l f\left[F_C\Lambda_s\cos\left(\omega t-\beta_1\right)+F_C\Lambda_s\cos\left(\omega t-2\alpha_m-\beta_1\right)\right. \\ \qquad \left.-F_{d2}\Lambda_0\cos 2\left(\omega t-\alpha_m-\varphi_2\right)\right]\cdots\cdots\cdots\text{composite fault} \end{cases} \tag{14}$$

3. Hybrid diagnosis method and verification

3.1. Method description

Based on the previously described theoretical study, a hybrid diagnosis method combining the mechanical characteristics, i.e., the stator and the rotor vibration characteristics, with the electrical features, i.e., the CCPB properties, can be proposed, as shown in **Table 1**.

As indicated in **Table 1**, the combining mechanical-electrical intersectional fault characteristics are unique for each fault. It will be obviously more advantageous than only employing either the vibration or the CCPB characteristics.

Running condition	Stator vibration	Rotor vibration	CCPB
Normal condition	2nd harmonic	—	—
Rotor eccentricity	2nd harmonic	2nd harmonic	1st harmonic
RISC	1st, 2nd, 3rd, 4th harmonics, comparing with normal condition, the 2nd decreased while 1st, 3rd, and 4th increased	1st harmonic	2nd harmonic
Composite fault	1st, 2nd, 3rd, and 4th harmonics, comparing with normal condition, the 1st, 3rd, and 4th increased	1st, 2nd, and 3rd harmonics	1st and 2nd harmonics

Table 1. Detailed criteria of the hybrid diagnosis method.

3.2. Experimental verification

The experimental verification is taken on a SDF-9 type fault simulating generator, in the National Key Lab of New Energy Electric Power System, P.R. China, as indicated in **Figure 6(a)**. The primary parameters of the generator are shown in **Table 2**.

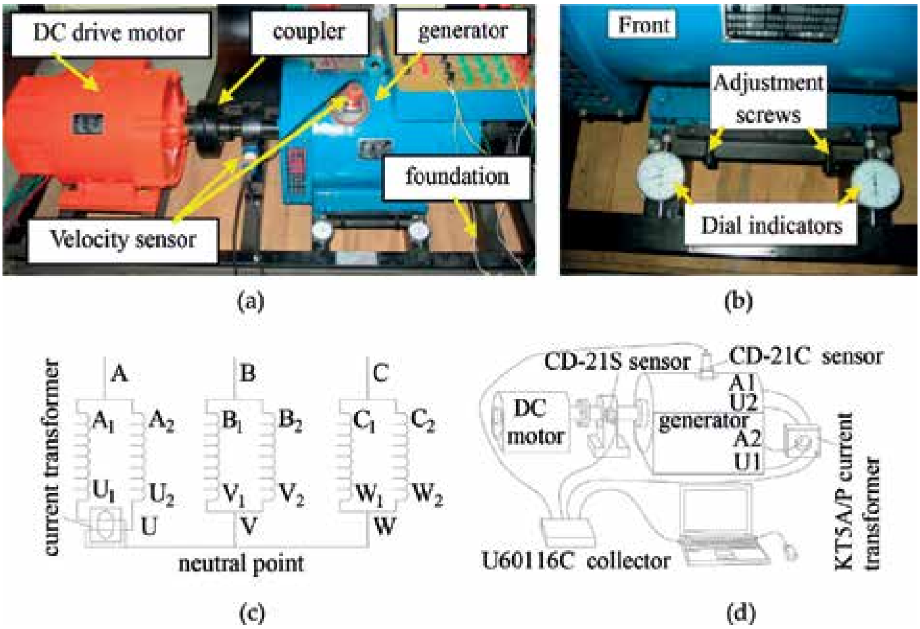


Figure 6. Experiment method of the fault simulating generator. (a) General outlook, (b) method to set rotor eccentricity, (c) method to test CCPB and (d) testing system of the experiment.

Parameters	Values
Rated capacity	7.5 kVA
Rated exciting current	1.5 A
Rated rotation speed	3000 r/min
Number of pole pairs	$p = 1$
Polar distance	$\tau = 252$ mm
Radial air-gap length	$g_0 = 0.8$ mm
Number of exciting slots per pole	6
Number of exciting turns peer pole	480
Number of stator slots	24
Number of turns in series per phase	100
Ratio of pitch to polar distance	$k_y = y/\tau = 0.83$
Pitch shortening factor	$k_p = 0.966$

Table 2. Primary parameters of study object.

The rotor of the generator is kept to the underframe by the bearing pedestal, while the stator can be moved along the horizontally radial direction by adjusting the screws; see **Figure 6(b)**. The movement can be controlled by two dial indicators, so that different fault degrees of rotor eccentricity can be simulated.

During the experiment, two velocity sensors are employed to test the stator vibration and the rotor vibration (see **Figure 6a**), while the CCPB is measured by a current transformer (see **Figure 6c**, the conductors of the two branches inversely cross the current transformer to get the current difference which is also the CCPB). The tested data are collected by a U60116C type collector and stored in the computer; see **Figure 6(d)**.

The stator vibration spectra for the four running conditions are indicated in **Figure 7**, while the rotor vibration spectra and the CCPB spectra for each performing case are illustrated in **Figures 8** and **9**, respectively. Theoretically, in normal condition, the stator should have only the second harmonic vibration component, and there should be no rotor vibrations or circulating currents. However, the experimental data show some differences. This is mainly caused by the asymmetry inside the generator. For example, the winding distribution in the generator may not be strictly symmetric. These initial values which should be zero in theory can be treated as the null drift of the generator system.

As indicated in **Figure 7**, it is shown that the four performing conditions will have different stator vibration characteristics. The occurrence of the rotor eccentricity will obviously increase the amplitude of the second harmonic, while the occurrence of RISC will decrease this

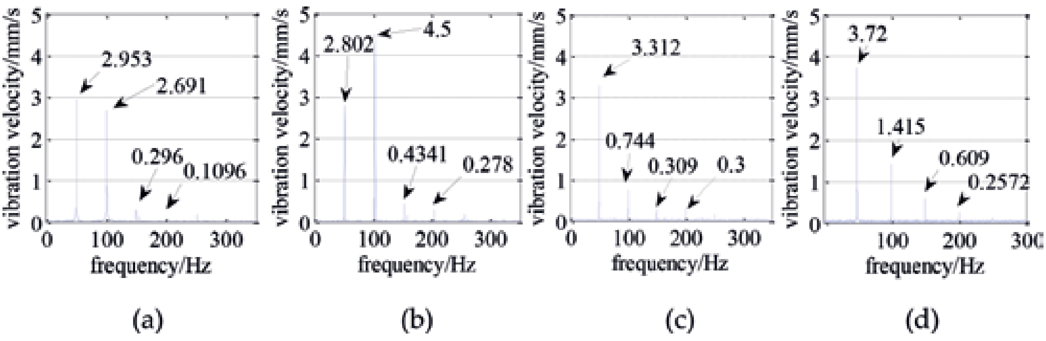


Figure 7. Stator vibration spectra under different conditions. (a) Normal condition, (b) 0.3 mm eccentricity, (c) 3% RISC and (d) composite fault.

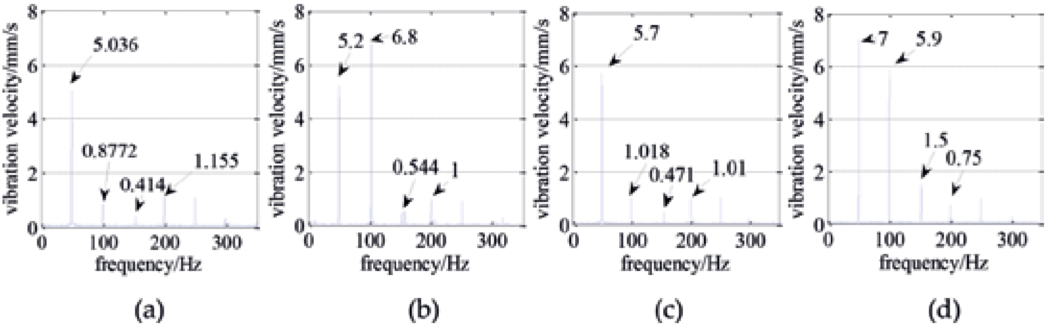


Figure 8. Rotor vibration spectra under different conditions. (a) Normal condition, (b) 0.3 mm eccentricity, (c) 3% RISC and (d) composite fault.

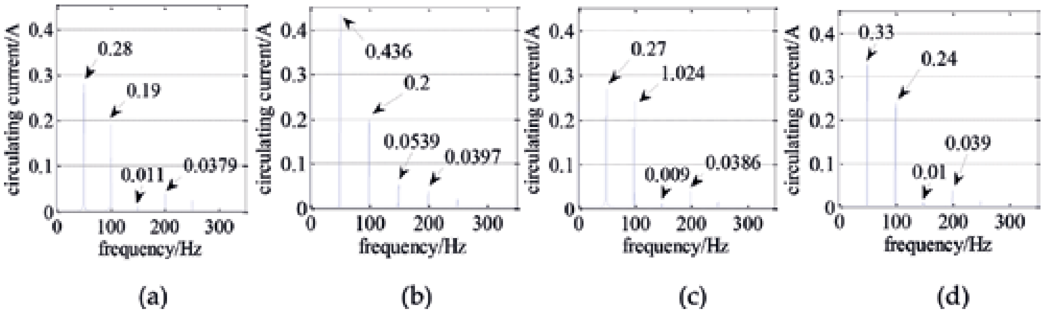


Figure 9. CCPB spectra under different conditions. (a) Normal condition, (b) 0.3 mm eccentricity, (c) 3% RISC and (d) composite fault.

harmonic but increase the first, third, and fourth harmonics. For the composite fault, the amplitude of the second harmonic is generally between the rotor eccentricity and RISC. The experimental results are consistent with the previously described theoretical analysis.

As indicated in **Figure 8**, the second harmonic vibration amplitude will be increased as the rotor eccentricity takes place, while the occurrence of RISC will increase the first harmonic amplitude. In the case of the composite fault, comparing with the normal condition, all of the first, second, and third harmonic amplitudes will be increased. This tendency to develop harmonic amplitude is in accordance with the theoretical result.

As indicated in **Figure 9**, the rotor eccentricity will mainly increase the first harmonic CCPB, while RISC will primarily increase the second harmonic CCPB. In the case of the composite fault, both the first and the second harmonic amplitudes of the CCPB will be enlarged. This experimental result still follows that of the previous theoretical study.

It is suggested from **Figures 7–9** that the stator vibration, the rotor vibration, and the CCPB will all vary due to different performing conditions. Combining the stator and the rotor vibration characteristics with the CCPB varying features, the mentioned four running conditions can be effectively and accurately identified. To confirm this, we have also carried out the experiments several times. And, by using the hybrid method proposed in this chapter, we correctly identified the running conditions each time.

4. Conclusions

In this chapter, we propose a new hybrid fault diagnosis method based on the intersectional mechanical-electrical characteristics. Primary conclusions drawn from the study are as follows.

1. Given the complex practical running condition, it is hard to identify the fault accurately only by either the stator vibration characteristics or the rotor vibration features.
2. In addition to the rotor and the stator vibration characteristics, the circulating current inside the parallel branches of the same phase is also an effective tool for condition monitoring and fault diagnosis.

3. By combining the rotor and stator vibration characteristics with the circulating current inside the parallel branches, the four running conditions, i.e., the normal condition, the single rotor eccentricity fault, the rotor interturn short circuit fault, and the composite fault composed of the rotor eccentricity and the rotor interturn short circuit, can be accurately identified due to the unique combining fault characteristics.

Acknowledgements

This work was in part supported by National Natural Science Foundation of China (51777074, 51307058), Natural Science Foundation of Hebei Province, China (E2015502013), and Chinese Fundamental Research Funds for the Central Universities (2018YQ03).

Conflict of interest

There is no conflict of interest regarding the publication of this work.

Author details

Yu-Ling He* and Yue-Xin Sun

*Address all correspondence to: heyuling1@163.com

Department of Mechanical Engineering, North China Electric Power University, Baoding, Hebei, China

References

[1] He Y-L, Ke M-Q, Tang G-J, Jiang H-C, Yuan X-H. Analysis and simulation on the effect of rotor interturn short circuit on magnetic flux density of turbo-generator. Journal of Electrical Engineering—Elektrotechnicky Casopis. 2016;**67**:323-333. DOI: 10.1515/jee-2016-0047

[2] Gandhi A, Corrigan T, Parsa L. Recent advances in modeling and online detection of stator interturn faults in electrical motors. IEEE Transactions on Industrial Electronics. 2011;**58**: 1564-1575. DOI: 10.1109/TIE.2010.2089937

[3] He Y-L, Wang F-L, Tang G-J, Ke M-Q. Analysis on steady-state electromagnetic characteristics and online monitoring method of stator inter-turn short circuit of turbo-generator. Electric Power Components and Systems. 2017;**45**:198-210. DOI: 10.1080/15325008.2016.1247387

[4] Tang G-J, Ke M-Q, He Y-L, Wang F-L. United electromagnetic characteristics and online monitoring method of static air-gap eccentricity of turbo-generator. Journal of Electrical Engineering and Technology. 2016;**11**:1614-1627. DOI: 10.5370/JEET.2016.11.6.1614

[5] Hao L, Wu J, Zhou Y. Theoretical analysis and calculation model of the electromagnetic torque of nonsalient-pole synchronous machines with interturn short circuit in field windings. IEEE Transactions on Energy Conversion. 2015;**30**:110-121. DOI: 10.1109/TEC.2014.2350 336

[6] He Y-L, Deng W-Q, Tang G-J. Impact of different static air-gap eccentricity forms on rotor UMP of turbogenerator. Mathematical Problems in Engineering. 2016;**2016**:1-13. https://www.hindawi.com/journals/mpe/2016/5284815/cta/. Open Access. DOI: 10.1155/2016/5284815

[7] Bruzzese C. Diagnosis of eccentric rotor in synchronous machines by analysis of split-phase currents—Part II: Experimental analysis. IEEE Transactions on Industrial Electronics. 2014;**61**:4206-4216. DOI: 10.1109/TIE.2013.2284554

[8] Zhu J-h, A-rui Q, Guo T. Branch voltage of a salient pole synchronous generator with eccentric rotor and skewed slots. Journal of Tsinghua University (Science and Technology). 2008;**48**:453-456

[9] Bruzzese C, Joksimovic G. Harmonic signatures of static eccentricities in the stator voltages and in the rotor current of no-load salient-pole synchronous generators. IEEE Transactions on Industrial Electronics. 2011;**58**:1606-1624. DOI: 10.1109/TIE.2010.2087296

[10] Doorsamy W, Abdallh AAE, Cronje WA, Dupré L. An experimental design for static eccentricity detection in synchronous machines using a Cramér–Rao lower bound technique. IEEE Transactions on Energy Conversion. 2015;**30**:254-261. DOI: 10.1109/TEC.2014.2347895

[11] Iamamura BAT, Le Menach Y, Tounzi A, Sadowski N, Guillot E. Study of static and dynamic eccentricities of a synchronous generator using 3-D FEM. IEEE Transactions on Magnetics. 2010;**46**:3516-3519. DOI: 10.1109/TMAG.2010.2043347

[12] Babaei M, Faiz J, Ebrahimi BM, Amini S, Nazarzadeh J. A detailed analytical model of a salient-pole synchronous generator under dynamic eccentricity fault. IEEE Transactions on Magnetics. 2011;**47**:764-771. DOI: 10.1109/TMAG.2011.2105498

[13] Dorrel DG. Experimental behaviour of unbalanced magnetic pull in 3-phase induction motors with eccentric rotors and the relationship with tooth saturation. IEEE Transactions on Energy Conversion. 1999;**14**:304-309. DOI: 10.1109/60.790874

[14] Wei-li L, Tang L, Xiao-cheng Z, Jia-ming G. Calculation and analysis of high-speed permanent magnetic generator unilateral magnetic force. In: Proceedings of the 11th International Conference on Electrical Machines and Systems; Oct 17–20; Wuhan, China. 2008

[15] Lin W, Cheung RW, Ma Z-y, Jiang-jun R, Ying P. Finite-element analysis of unbalanced magnetic pull in a large hydro-generator under practical operations. IEEE Transactions on Magnetics. 2008;**44**:1558-1561. DOI: 10.1109/TMAG.2007.916023

[16] Joksimovic GM. Dynamic simulation of cage induction machine with air gap eccentricity. IEE Proceedings—Electric Power Applications. 2005;**152**:803-811

[17] Tabatabaei I, Faiz J, Lesani H, Nabavi-Razavi MT. Modeling and simulation of a salient-pole synchronous generator with dynamic eccentricity using modified winding function theory. IEEE Transactions on Magnetics. 2004;**40**:1550-1555. DOI: 10.1109/TMAG.2004.826611

[18] Faiz J, Ojaghi M. Unified winding function approach for dynamic simulation of different kinds of eccentricity faults in cage induction machines. IET Electric Power Applications. 2009;**3**:461-470. DOI: 10.1049/iet-epa.2008.0206

[19] Dorrell DG, Salah A. Detection of rotor eccentricity in wound rotor induction machines using pole-specific search coils. IEEE Transactions on Magnetics. 2015;**51**:1-4. DOI: 10.1109/TMAG.2015.2443711

[20] Sulla F, Svensson J, Samuelsson O. Symmetrical and unsymmetrical short-circuit current of squirrel-cage and doubly-fed induction generators. Electric Power Systems Research. 2011;**81**:1610-1618. DOI: 10.1016/j.epsr.2011.03.016

[21] Klontz KW, TJE M, MI MG, Karmaker H, Zhong P. Short-circuit analysis of permanent-magnet generators. IEEE Transactions on Industry Applications. 2011;**47**:1670-1680. DOI: 10.1109/TIA.2011.2154370

[22] Albright DR. Inter-turn short-circuit detector for turbine-generator rotor windings. IEEE Transactions on Power Apparatus and Systems. 1971;**PAS-90**:478-483. DOI: 10.1109/TPAS.1971.293048

[23] Fiser R, Lavric H, Bugeza M, Makuc D. FEM modeling of inter-turn short-circuits in excitation winding of turbo-generator. Przeglad Elektrotechniczny. 2011;**87**:49-52

[24] Li GJ, Hloui S, Ojeda J, Hoang E, Lecrivain M, Gabsi M, Zhu ZQ. Excitation winding short-circuits in hybrid excitation permanent magnet motor. IEEE Transactions on Energy Conversion. 2014;**29**:567-575. DOI: 10.1109/TEC.2014.2322194

[25] Shuting W, Yonggang L, Heming L, Guiji T. The analysis of generator excitation current harmonics on stator and rotor winding fault. In: IEEE International Symposium on Industrial Electronics; 9–13 July; Montreal, Que., Canada. 2006

[26] Wallin M, Lundin U. Dynamic unbalanced pull from field winding turn short circuits in hydropower generators. Electric Power Components and Systems. 2013;**41**:1672-1685. DOI: 10.1080/15325008.2013.835360

[27] Yonggang L, Guowei Z, Yucai W, Heming L. Impact of rotor inter-turn short-circuit on generator rotor force. Electrical Information and Mechatronics and Applications. 2012; **143-144**:125-131. DOI: 10.4028/www.scientific.net/AMM.143-144.125

[28] Ramirez-Nino J, Pascacio A. Detecting interturn short circuits in rotor windings. Computer Applications in Power, IEEE. 2001;**14**:39-42. DOI: 10.1109/67.954526

[29] Campbell SR, Kapler J, Sasic M, Stone GC. Detection of rotor winding shorted turns in turbine generators and hydrogenerators. In: Cigre 2010 Session; 22–27 August; Paris, France. 2010

[30] Biet M. Rotor faults diagnosis using feature selection and nearest neighbors rule: Application to a turbogenerator. IEEE Transactions on Industrial Electronics. 2013;**60**:4063-4073. DOI: 10.1109/TIE.2012.2218559

[31] Khezzar A, Kaikaa MY, El Kamel Oumaamar M, Boucherma M, Razik H. On the use of slot harmonics as a potential indicator of rotor bar breakage in the induction machine. IEEE Transactions on Industrial Electronics. 2009;**56**:4592-4605. DOI: 10.1109/TIE.2009.2030819

[32] Wu Q, Nandi S. Fast single-turn sensitive stator interturn fault detection of induction machines based on positive- and negative-sequence third harmonic components of line currents. IEEE Transactions on Industry Applications. 2010;**46**:974-983. DOI: 10.1109/TIA.2010.2045329

Section 3

Power Electronics

Chapter 4

A Comparative Study on Some Fault Diagnosis Techniques in Three-Phase Inverter Fed Induction Motors

Bilal Djamal Eddine Cherif, Azeddine Bendiabdellah,
Mokhtar Bendjebbar and Laribi Souad

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.79960

Abstract

The growing importance of power conversion systems and their dependency on the performance and reliability of static converters has motivated extensive research efforts in this field. A variety of different techniques have been applied to detect open-circuit faults in power converters. The present chapter is focusing on the techniques of detection and localization of open-circuit faults in a three phase voltage source inverter fed induction motor. A comparative study is carried out between different detection techniques: the Park current vectors and its enhancement by using the polar coordinates, the mean value of the currents, the stator current spectrum analysis and the measurement of the current drop. The aim of this comparison is to investigate the relative strengths and weaknesses of the different techniques and evaluate the performance of each detection technique studied. The comparison study focuses on the time detection, the localization ability and the hardware aspect. To validate these techniques, an experimental setup is developed in our diagnostic group laboratory which consists of a two-level voltage source inverter controlled by a *DSPACE-1104*, Card to generate the PWM vector control of the induction motor. The obtained simulation and experimental results illustrate well the detection feasibility of each technique as well as the benefits and merits of the performed comparative study.

Keywords: induction motor, inverter, open-circuit fault, detection, localization, Park's vectors, polar coordinates, mean value, frequency spectrum, current drop

1. Introduction

The applications of electronics were for a long time limited to the technique of high frequencies. The possibilities of application were limited by the unreliability of the electronic elements available at that time. This reliability was insufficient to meet the high demands of new applications in the industrial field. It was only after the development of special electronic power components of higher reliability and more limited tolerance that new techniques could be envisaged, thus creating a new branch of electronics called power electronics.

One area where power electronics is widely used today is the industrial applications related to machine-static power converter associations, particularly the industrial variable speed AC drives. These drives are mainly used in which the machines are connected through a static power electronic converter usually a three-phase inverter. It is estimated that about 38% of the faults in these industrial drives are due to failures of the supply system. Some uses of the machine-static power converter sets do not tolerate untimely failures, mechanical or electrical failures at the machine or related to static power converter failure. An industrial survey [1] conducted in 2011, comes to the conclusion that 93% of respondents stated that reliability is a paramount issue in the field of static power electronic converters. The faults of the static power converters have various causes; they can be related to the open-circuit faults of the IGBT switches for example. This type of malfunction induces damage constraints for the production system if the personnel are not notified and a nuisance shutdown can eventually result.

The growing interest of manufacturers in the maintenance of electrical drives justifies the emphasis placed on research into the diagnosis of machine-static power converter associations. The complexity of the systems involved and the necessary approach from the new angle of diagnosis today require a preliminary work of detection/diagnosis of the faults of the machine-converter association.

Several researchers have studied the behavior of static power converters with internal failure, focusing particularly on the open-circuit fault of an IGBT switch. Such a fault can lead to secondary faults in other converter components that can result in high repair costs [2].

Authors [3, 4] propose the Park vector technique, the principle of this technique is based on the tracking of the current trajectory of Park (i_d, i_q). In the healthy case, the trajectory takes a circle shape and in the case of an IGBT switch open-circuit fault, the circle becomes a semicircle.

The position of this trajectory in the *(d-q)* frame makes it possible to calculate the intervals of the angles of the fault to localize the faulty IGBT. Other researchers [5–8] have proposed the Park.

Average current (i_{dmean}, i_{qmean}) technique to calculate the exact open-circuit fault angle in order to identify the faulty IGBT switch. Authors [9–11] have proposed the technique based on the spectral analysis of the stator currents. This technique is based on the study of the harmonic analysis of each phase current. The amplitude and argument of each harmonic can be used in detecting and localizing the faults. The analysis of the first harmonics shows that the difference between the healthy state and the open-circuit fault case resides in the zero-order harmonics which signifies the presence of the DC component in the signal. The argument of the harmonic zero with respect to the fundamental makes it possible to know the type of fault; on the other hand, the argument of this harmonic also makes it possible to know the faulty IGBT switch either the high or the lower one. The authors [12] combined normalized standard currents with additional diagnostic variables for one or more IGBT switch open-circuit faults. The diagnostic alarms were performed by the Boolean output signals. In the paper presented by [13], the same authors proposed another extension based on the use of fuzzy logic symptoms. The author [14] proposes the Clarke technique followed by the polarity of the trajectory slope in the complex α-β frame identify the faulty IGBT switch.

The present chapter is focusing on techniques of detection and localization of IGBT switch open-circuit faults in a three phase voltage source inverter fed induction motor. A comparative study is carried out between different detection techniques: the Park current vectors and its enhancement by using the polar coordinates, the mean value of the currents, the stator current spectrum analysis and the measurement of the current drop. The comparison study aims at exhibiting the relative strengths and weaknesses of the different techniques and at assessing each detection technique in terms of its performance; that is the time detection and the localization ability; as well as in terms of hardware; that is the number of current sensors required for IGBT switch open-circuit fault detection. To validate these techniques, an experimental setup is developed in our diagnostic group laboratory which consists of a two-level voltage source inverter controlled by a *DSPACE-1104* Card to generate the *PWM* vector control of the induction motor. The obtained simulation and experimental results illustrate well the detection feasibility of each technique as well as the benefits and merits of the performed comparative study.

2. Space voltage vector and switching states for the case of both healthy and faulty inverter

Figure 1 shows the general structure of a three phase two-level voltage source inverter feeding an induction motor.

This inverter is controlled by the *PWM* vector control strategy. For each leg of the inverter, there are two possible states:

State 1: The higher switch Kx ($x = 1, 2$ *or* 3) is closed, while the lower switch K_2x ($x = 1, 2$ *or* 3) is open. The output voltage relative to the neutral of the dc source is V_{dc}.

State 0: The lower switch Kx ($x = 1, 2$ *or* 3) is closed, while the higher switch K_1x ($x = 1, 2$ *or* 3) is open. The output voltage relative to the neutral of the dc source is *0v.*

It is to note that the faulty inverter used in this chapter is defined as an inverter with one of its IGBT switch exhibiting an open-circuit fault.

The experimental setup used in this chapter is depicted in the photo of **Figure 2**. It includes a three-phase squirrel-cage induction motor fed by a three-phase two-level voltage source inverter.

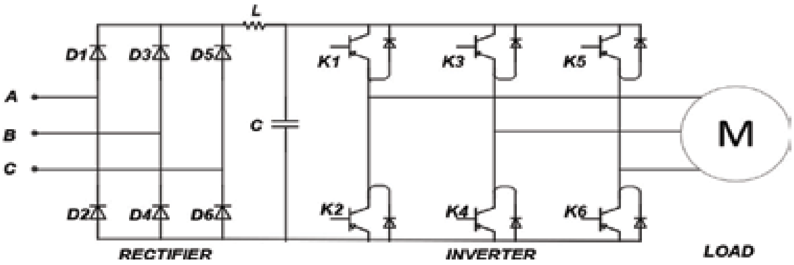


Figure 1. The structure of a three phase two-level inverter.

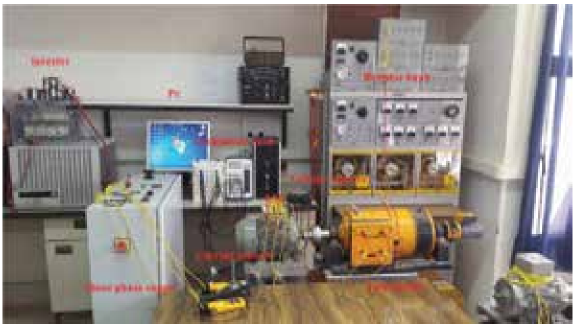


Figure 2. Experimental setup.

The detailed characteristics of the motor are given in the appendix. Furthermore, the motor is mechanically coupled to a DC generator supplying a bank of resistors which allows varying the load torque. Moreover, the measuring system includes three current Hall Effect sensors and three voltage sensors and a DSPACE 1104 acquisition card to generate pulses for triggering the IGBT's gates in the inverter. The whole set is connected to a computer for visualizing and analyzing the processed sensed signal [14].

Because of the randomness of the measured signals and for a reliable analysis, 05 acquisitions are performed for each case. The acquisition time used is T_{acq} = 20s. To study the effect of the load on the induction motor signals, the following mode of operation is considered; the rated load operation with a rated current of 7A and an estimated torque of 20 Nm and a frequency of sampling Fe = 1.5kHz.

3. Fault detection techniques for a faulty inverter

During its operation, the inverter is subjected to various internal and external constraints resulting in its failure; especially those failures related to the IGBT semiconductor switches because of their fragility. Two types of faults are usually linked with the inverter and can be reported in **Table 1** as follows:

Faults	Description
Short-circuit	Short-circuit faults affecting the IGBT switches are the most serious faults. In the presence of such a fault, the current reaches limits which can cause the fusion of its chip or its connection. If the detection of this type of fault does not occur rapidly (less than 10 μs), then the IGBT switch which is still active on the same leg undergoes the same phenomenon and so the whole inverter leg is shorted.
Open-circuit	Open-circuit faults affecting the IGBT switches may occur when, for any reason, the IGBT is disconnected, is damaged, or had a problem in its grid control signal. This type of fault is very difficult to perceive directly because the motor can continue to operate but with a degradation of its performance due to the occurrence of fluctuations in the mechanical parameters (speed and torque) as well as an imbalance of the currents where the currents of the other two healthy legs take high values to maintain the average torque and the speed. The starting of the motor in the presence of this type of fault cannot always be ensured.

Table 1. Description of inverter faults.

Number	Methods for open-circuit IGBT switch faults
1	Technique based on the Park vectors and enhanced Park vectors with polar coordinates
2	Technique based on the mean value of the currents
3	Technique based on the stator current spectrum analysis
4	Technique based on the measure of the current drop

Table 2. Fault diagnosis used techniques.

In this chapter section, different techniques for fault detection and localization of the inverter IGBT switches open-circuit fault are presented in the summarized **Table 2** and thoroughly discussed.

3.1. Technique based on the Park vectors

The Park vectors technique is based on the tracking of the trajectory of the current vector. Indeed, in normal conditions (without fault), the trajectory of the current vector in the *d-q* frame is a circle. The circle becomes a semicircle when an IGBT open-circuit fault of an inverter leg occurs. The position of this semicircle in the *d-q* frame allows identifying the faulty switch as developed by [10]. Applying the Park transformation on the three phase currents (i_a, i_b, i_c) results in two currents (i_d, i_q) in the *d-q* frame. The current is expressed by the following mathematical system:

$$\begin{cases} \boldsymbol{i}_{ds} = {}^{2}\!/_{3}\,\boldsymbol{i}_{as} - {}^{1}\!/_{3}\,\boldsymbol{i}_{bs} - {}^{1}\!/_{3}\,\boldsymbol{i}_{cs} \\ \boldsymbol{i}_{qs} = {}^{1}\!/_{\sqrt{3}}\left(\boldsymbol{i}_{bs} - \boldsymbol{i}_{cs}\right) \end{cases} \tag{1}$$

Its slope is used to identify the faulty leg. As mentioned above, the extraction of information of any faulty leg can be obtained from the angle between the current vector and the *d-q* frame using the following equation:

$$\boldsymbol{\theta}_l = \mathbf{arctan}\left({}^{i_{qs}}\!/_{i_{ds}}\right) \tag{2}$$

where $\boldsymbol{\theta}_l$ is the inverter leg fault angle.

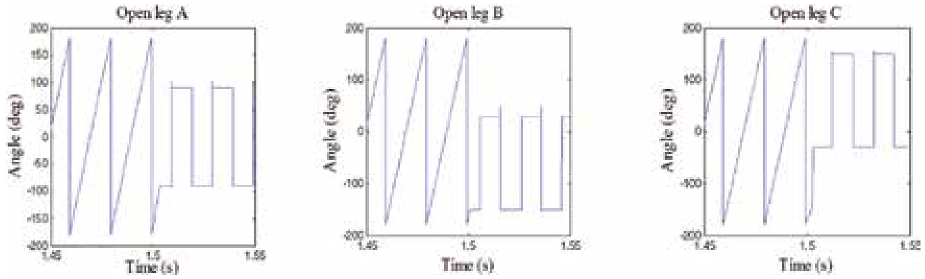


Figure 3. Ideal shapes of the leg fault angle $\boldsymbol{\theta_l}$ (simulation results).

Figure 3 depicts the angle of the faulty leg when applying an open-circuit fault at *t = 1.5 s*.

In reference to **Figure 3**, the leg fault angle $\boldsymbol{\theta_l}$ is used to identify the faulty leg of the inverter. For the case of a healthy inverter, the angle values are always π or -π. Any angle values different from π or -π are therefore considered as an indication confirming faulty legs presence. The various faulty inverter legs corresponding to the various fault angle values are expressed in **Table 3**.

The Park trajectory slope is used to identify the faulty switch. As mentioned above, the extraction of information of any faulty switch can be obtained from the angle between the mean current vector and the *d-q* frame using the following equation:

$$\boldsymbol{\theta_K} = \arctan\left({}^{i_{qsmean}}\!/_{i_{dsmean}} \right) \tag{3}$$

where $\boldsymbol{\theta_K}$ is the inverter switch fault angle.

Figure 4 depicts the angle of the faulty switch when applying an open-circuit fault at *t* = 1.5 s.

Figure 4 shows the fault angles switch $\boldsymbol{\theta_K}$. The various faulty inverter switches corresponding to the various fault angle values are expressed in **Table 4**.

For example for the case of a faulty inverter switch K_1 corresponds the angle intervals *[0, π/2]* or *[3π/2, 2π]*.

Figure 5 depicts the trajectories of the currents for both healthy and faulty inverters.

Faulty leg	IGBT switches	Leg fault angle θ_l
leg A	K_1, K_2	*π/2 or –π/2*
leg B	K_3, K_4	*5π/6 or –π/6*
leg C	K_5, K_6	*5π/6 or –π/6*

Table 3. Current vector position as function of $\boldsymbol{\theta_l}$.

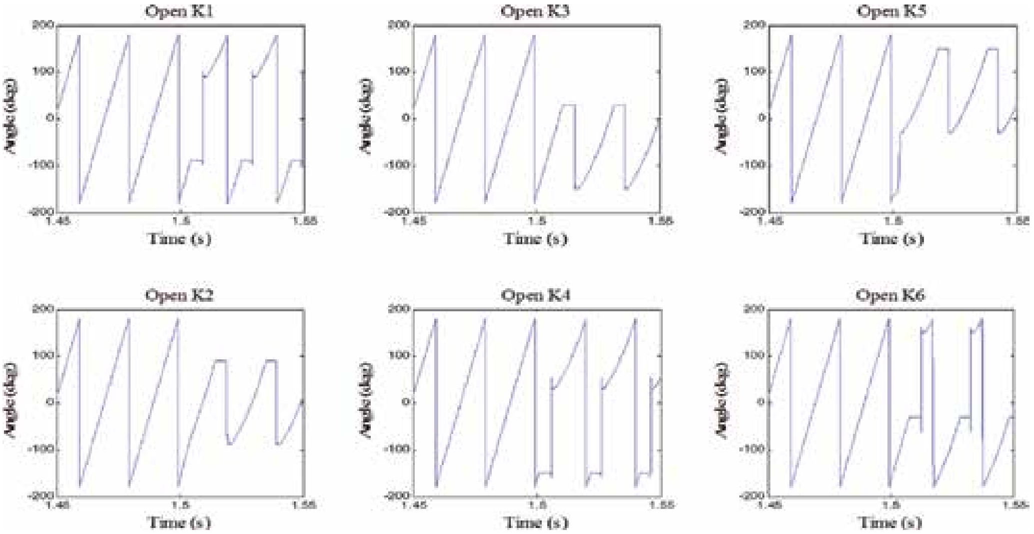


Figure 4. Ideal shapes of the IGBT switch open-circuit fault angle θ_K (simulation result).

IGBT	**Fault angles of inverter switch** θ_K
K_1 open	*[0, π/2] or [3π/2, 2π]*
K_2 open	*[π/2, 3π/2]*
K_3 open	*[π/6, 7π/6]*
K_4 open	*[0, π/6] or [7π/6, 2π]*
K_5 open	*[5π/6, 11π/6]*
K_6 open	*[0, 5π/6] or [11π/6, 2π]*

Table 4. Open-circuit switch fault angles.

3.1.1. Enhanced Park vectors with polar coordinates

A polar coordinates calculation is proposed associated with the technique of Park vectors in order to enhance the inverter IGBT switch fault localization. This section focuses on the localization of each faulty IGBT switch by calculating the borders of each trajectory as well as the fault current vector. This calculation is carried out by proposing the use of the polar coordinates diagram as shown by **Figure 6**.

The mathematical model based on the polar coordinates is related to the trajectory and the angle θ_{icf} of the specified faulty IGBT. Consequently, a change in the shape of the trajectory and the angle indicates the occurrence of a fault condition. This represents an indicator for the localization of the faulty IGBT switch. The angle is calculated from Eqs. 4–6 as follows:

$$\theta_{icf} = \sum_{i=0}^{N} {}^{\pi r^2 \rho}\big/_{360} \tag{4}$$

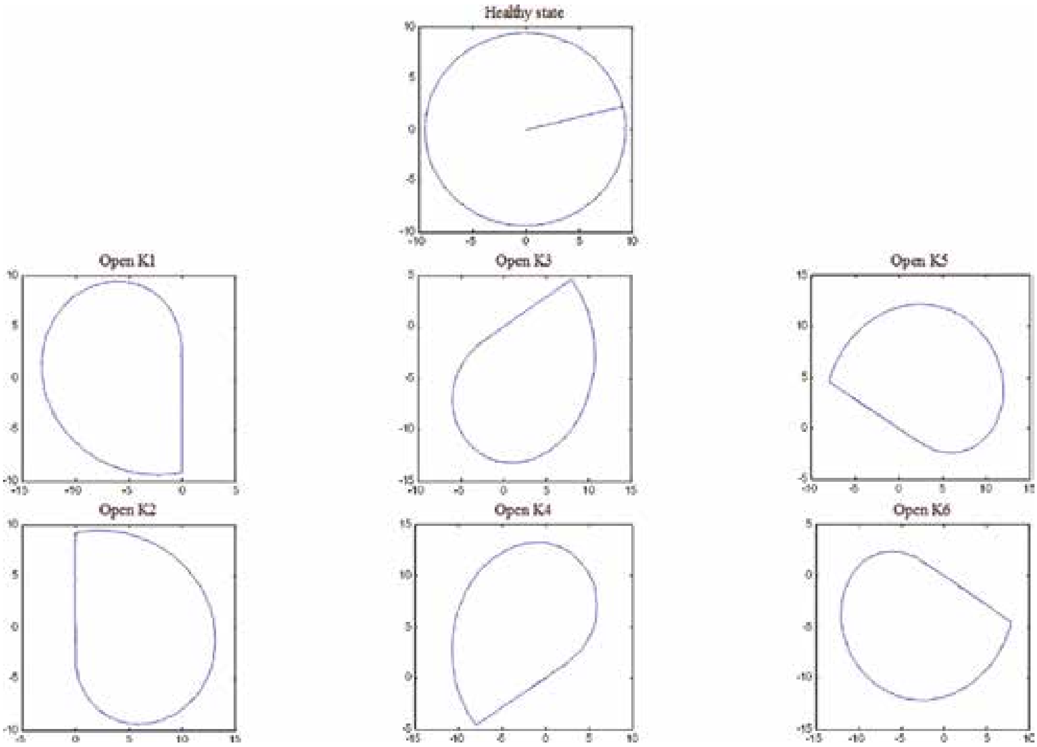


Figure 5. Currents trajectories of healthy and faulty inverter (simulation results).

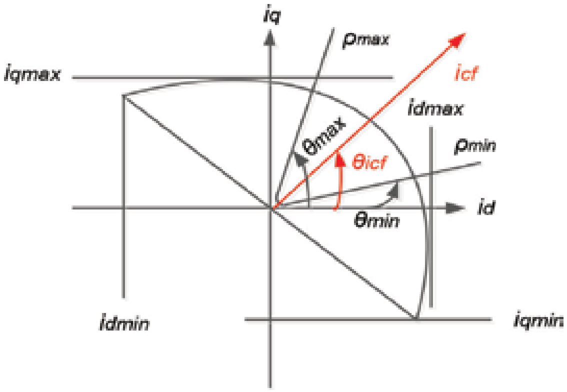


Figure 6. Proposed schematic geometry of polar coordinates.

The radius of this trajectory can be calculated from the following equation:

$$r=\sqrt{{i_{ds}}^{2}+{i_{qs}}^{2}} \tag{5}$$

$$\rho=\rho_{max}-\rho_{min} \tag{6}$$

IGBT	*Fault current vector* i_{cf}
K_1 open	$i_{cf} = 6.5\angle\pi$
K_2 open	$i_{cf} = 6.5\angle 2\pi$
K_3 open	$i_{cf} = 4.95\angle 5\pi/3$
K_4 open	$i_{cf} = 4.95\angle 2\pi/3$
K_5 open	$i_{cf} = 5.3\angle\pi/3$
K_6 open	$i_{cf} = 5.3\angle 4\pi/3$

Table 5. Fault current vector for each faulty switch.

The d-q currents at the center of the trajectory i_{dc} and i_{qc} can be computed using the maximum and minimum of the currents vectors as follows:

$$\begin{cases} i_{dsc} = \tfrac{1}{2}\left(i_{dsmax} + i_{dsmin}\right) \\ i_{qsc} = \tfrac{1}{2}\left(i_{qsmax} + i_{qsmin}\right) \end{cases} \tag{7}$$

For the case of a faulty IGBT switch, the current vector is given by the following equation:

$$i_{cf} = i_{dsc} + j i_{qsc} \tag{8}$$

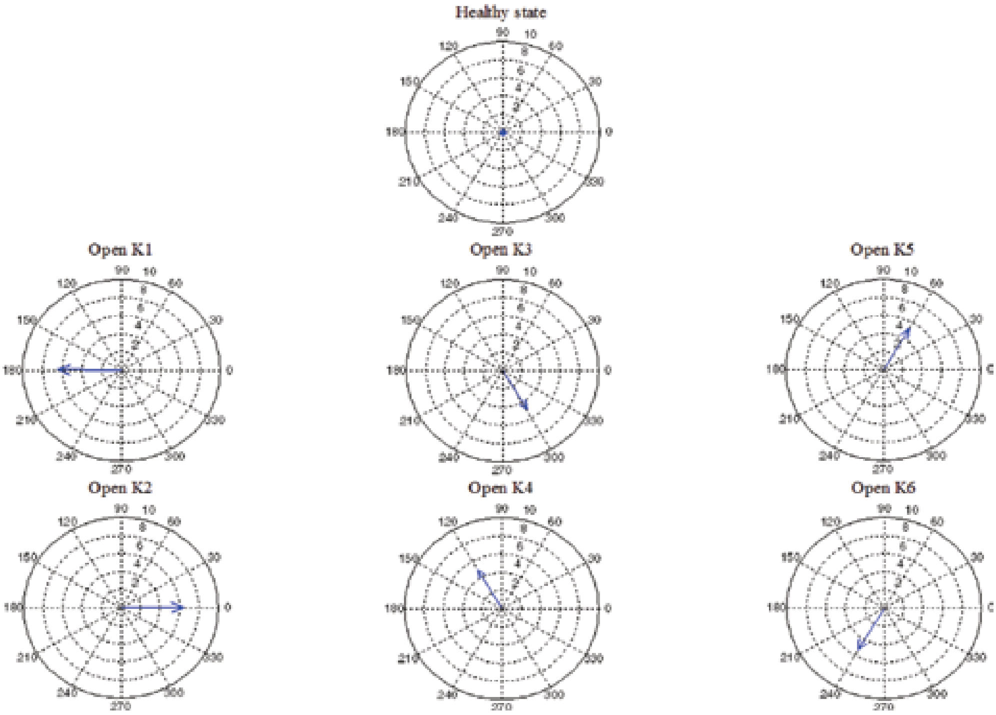


Figure 7. Graphical representation of fault current vectors (simulation results).

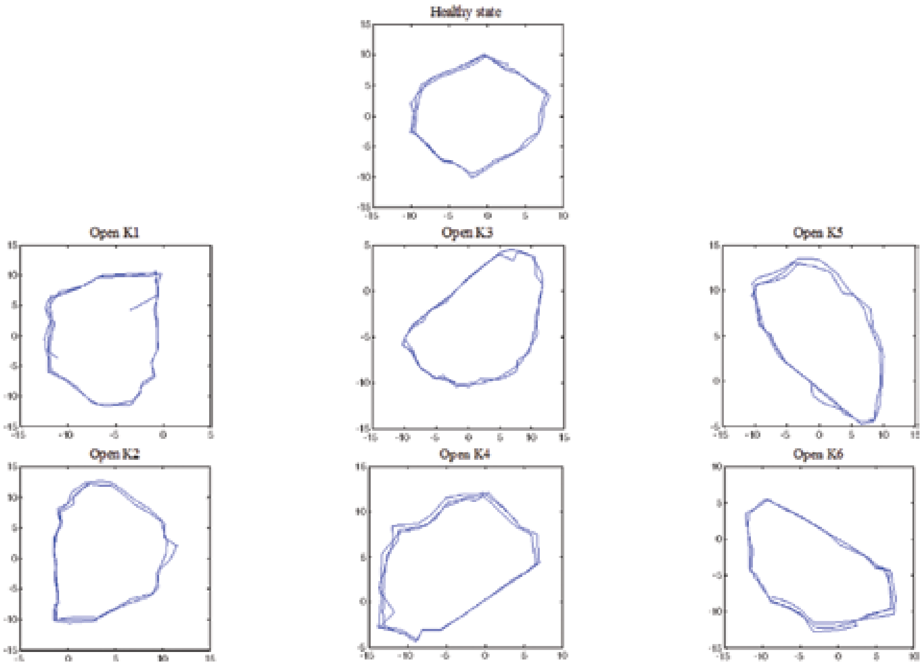


Figure 8. Currents trajectories of healthy and faulty inverter (experimental result).

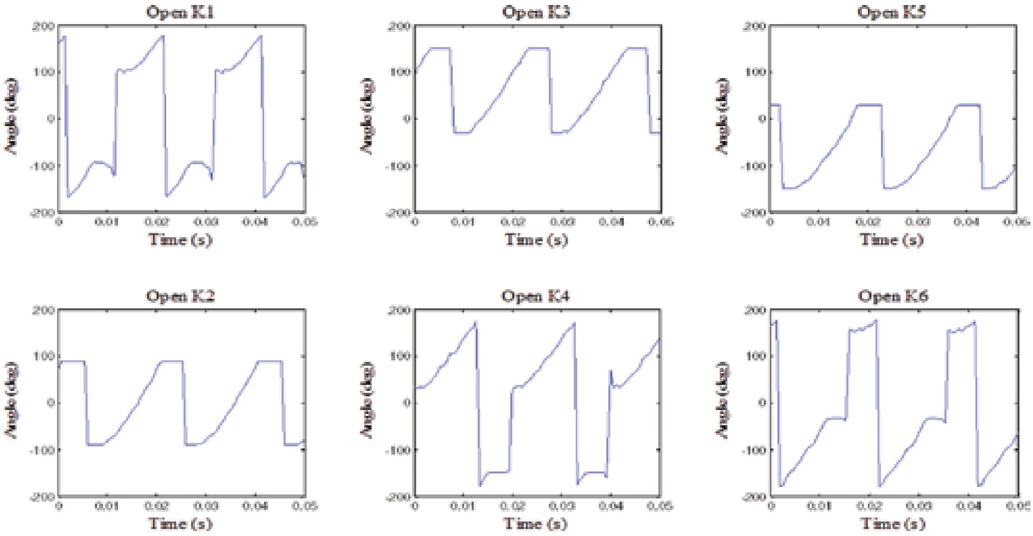


Figure 9. Ideal shapes of the IGBT switch open-circuit fault angle θ_K (experimental results).

From this calculation, the open-circuit fault of each IGBT switch can be localized and **Table 5** summarizes the computation of the fault angle values and the fault current vectors for the case of each faulty IGBT switch.

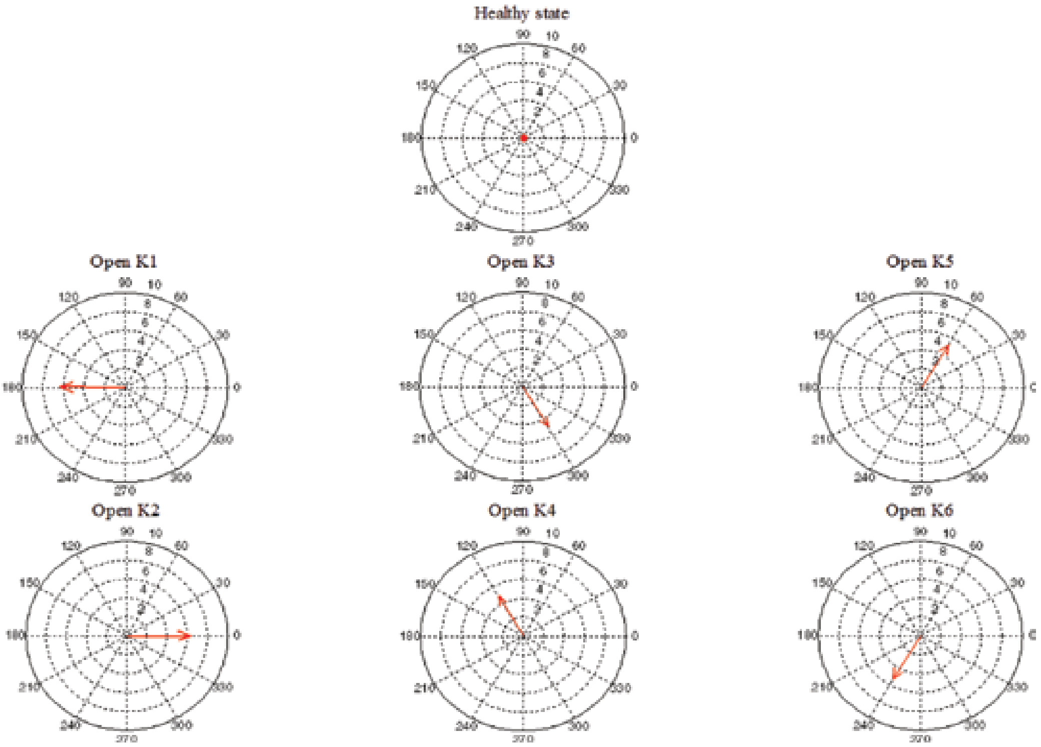


Figure 10. Graphical representation of fault current vectors (experimental results).

A significant advantage of the polar coordinates calculation is that it enables to yield the exact value of the angle for inverter faults detection while other existing techniques gives rather an interval of angles.

Figure 7 shows the fault current vector $\boldsymbol{i_{cf}}$ when the inverter is operating under both healthy and faulty conditions. For the case of a healthy inverter, the average amplitude of the fault current vector is zero. If a failure occurs, the magnitude of the fault current vector becomes non-zero. The faulty IGBT is identified by the phase angle and the fault current vector magnitude.

3.1.2. Experimental results for healthy and faulty inverters using the Park vectors technique with polar coordinates

See **Figures 8–10**.

4. Technique based on the mean value of the currents

This technique uses the mean phase current value for fault detection. A fault in a semiconductor switch can produce offsets in the currents of the electrical machine phases. This diagnostic technique is to calculate the mean values of these currents from which the fault can be detected. A current threshold is defined in order to distinguish between the open-circuit faults in a semiconductor switch.

The mathematical model of this technique is illustrated by the following steps:

1st step: Extraction of the three currents of the stator (i_{as}, i_{bs}, i_{cs}) as follows:

$$\begin{cases} I_{as} = 0 \\ I_{bs} = I\cos\left(\omega t + \varphi - \dfrac{2\pi}{3}\right) \\ I_{cs} = Icos\left(\omega t + \varphi - \dfrac{4\pi}{3}\right) \end{cases} \tag{9}$$

2nd step: Calculation of the mean value of the three stator currents:

$$\begin{cases} I_{amean} = sum\left({}^{I_{as}}/_{length(i_a)}\right) \\ I_{bmean} = sum\left({}^{I_{bs}}/_{length(i_b)}\right) \\ I_{cmean} = sum\left({}^{I_{cs}}/_{length(i_c)}\right) \end{cases} \tag{10}$$

where length (Ia,b,c): number of point the three current signals *Ia, Ib, Ic.*

After the introduction of a fault in the IGBT of the inverter, one can easily observe the change in the form of the stator currents and also their mean values. This technique allows us therefore to easily identify and localize the faults which are expressed by the following equation:

$$I_{mean} = {}^{(I_{amean} + I_{bmean} + I_{cmean})}/_{3} \tag{11}$$

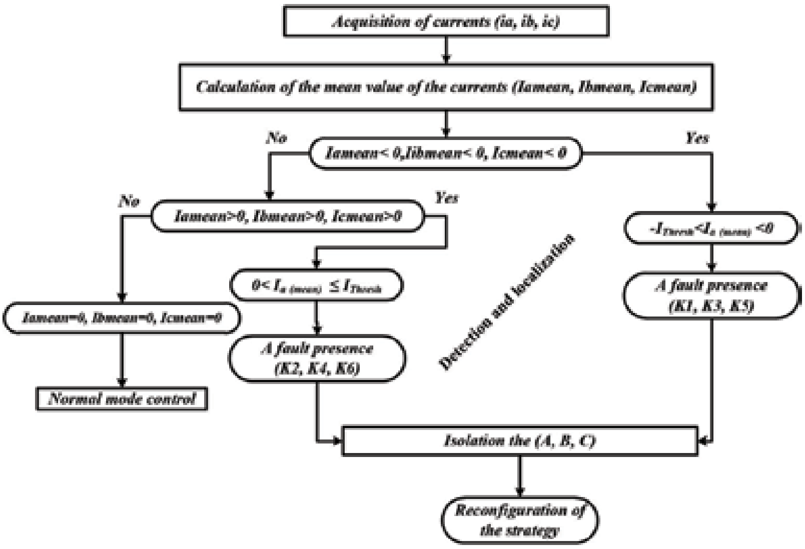

Acquisition of currents (ia, ib, ic)
Calculation of the mean value of the currents (Iamean, Ibmean, Icmean)
No
Iamean< 0,Iibmean< 0, Icmean< 0
Yes
No
Iamean>0, Ibmean>0, Icmean>0
Yes
-I_Thresh<I_a (mean) <0
0< I_a (mean) ≤ I_Thresh
A fault presence (K1, K3, K5)
Iamean=0, Ibmean=0, Icmean=0
A fault presence (K2, K4, K6)
Detection and localization
Normal mode control
Isolation the (A, B, C)
Reconfiguration of the strategy

Figure 11. Detection algorithm of faulty inverter using the mean value of stator currents technique.

State	Phase *A*	Phase *B*	Phase *C*
Healthy	I_{Thresh}, (I_{Thresh} =0)	I_{Thresh}, (I_{Thresh} =0)	I_{Thresh}, (I_{Thresh} =0)
K_1 open	$-I_{Thresh} < I_{amean} < 0$, ($I_{amean} = -1.5$)	$I_{bmean} \geq 0$, ($I_{bmean} = 0$)	$I_{cmean} \geq 0$, ($I_{cmean} = 0$)
K_3 open	$I_{amean} \geq 0$, ($I_{amean} = 0$)	$-I_{Thresh} < I_{bmean} < 0$, ($I_{bmean} = -1.5$)	$I_{cmean} \geq 0$, ($I_{cmean} = 0$)
K_5 open	$I_{amean} \geq 0$, ($I_{amean} = 0$)	$I_{bmean} \geq 0$, ($I_{bmean} = 0$)	$-I_{Thresh} < I_{cmean} < 0$, ($I_{cmean} = -1.5$)
K_2 open	$0 < I_{amean} \leq I_{Thresh}$, ($I_{amean} = 1.5$)	$I_{bmean} \leq 0$, ($I_{bmean} = 0$)	$I_{cmean} \leq 0$, ($I_{cmean} = 0$)
K_4 open	$I_{amean} \leq 0$, ($I_{amean} = 0$)	$0 < I_{bmean} \leq I_{Thresh}$, ($I_{bmean} = 1.5$)	$I_{cmean} \leq 0$, ($I_{cmean} = 0$)
K_6 open	$I_{amean} \leq 0$, ($I_{amean} = 0$)	$I_{bmean} \leq 0$, ($I_{bmean} = 0$)	$0 < I_{cmean} \leq I_{Thresh}$, ($I_{cmean} = 1.5$)

Table 6. Characteristics of various types of fault of the inverter.

The open-circuit fault detection algorithm that will be applied in this technique is based on the calculated mean value of the currents and can be described by the following algorithm as shown in **Figure 11** (**Table 6**).

4.1. Simulation results for healthy and faulty inverter using the technique of the mean value of the currents

See **Figures 12** and **13**.

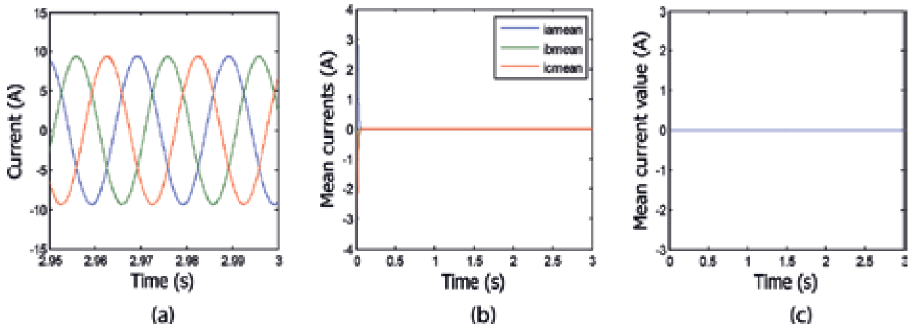


Figure 12. Current characteristics for a healthy inverter (simulation results). (a) Three stator currents (I_a, I_b, I_c), (b) Mean values of three stator currents (I_{amean}, I_{bmean}, I_{cmean}), (c) Mean value of current I_{mean}.

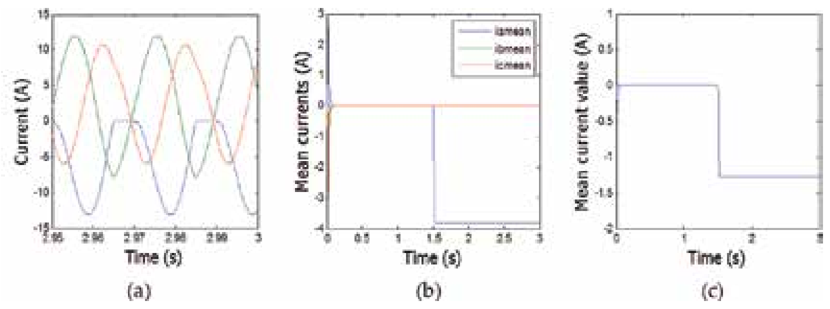


Figure 13. Currents trajectories for a faulty inverter. (a) Three stator currents (I_a, I_b, I_c), (b) Mean values of three stator currents (I_{amean}, I_{bmean}, I_{cmean}), (c) Mean value of current I_{mean}.

4.2. Experimental results for healthy and faulty inverter using the technique of the mean value of the currents

See **Figures 14** and **15**.

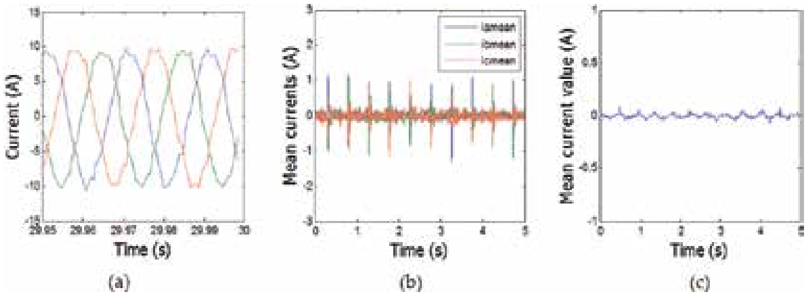


Figure 14. Current characteristics for a healthy inverter. (a) Three stator currents (I_a, I_b, I_c), (b) Mean values of three stator currents (I_{amean}, I_{bmean}, I_{cmean}), (c) Mean value of current I_{mean}.

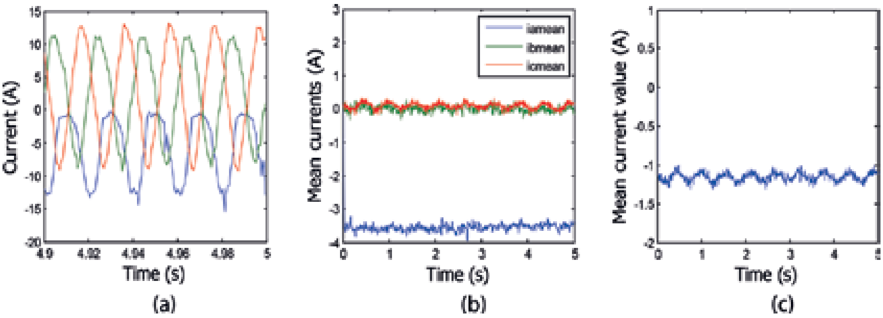


Figure 15. Currents trajectories for a faulty inverter (experimental results). (a) Three stator currents (I_a, I_b, I_c), (b) Mean values of three stator currents (I_{amean}, I_{bmean}, I_{cmean}), (c) Mean value of current I_{mean}.

5. Technique based on the stator current spectrum analysis

This technique is based on the study of the harmonic analysis of each phase current. The amplitude and the argument of each harmonic may be used in the detection and location of the faults [15].

5.1. Simulation results for healthy and faulty inverter using the stator current spectrum analysis

In what follows we will present the simulation for both the healthy state and the inverter with open-circuit switch fault state. **Figure 16** shows the harmonic spectrum of the current for the healthy state.

For an open-circuit K_1 IGBT fault, the harmonic spectrum for each phase is shown as in **Figure 17**. Note that the occurrence of the zero harmonic (i.e., the presence of the DC component) indicates the presence of a fault in K_1.

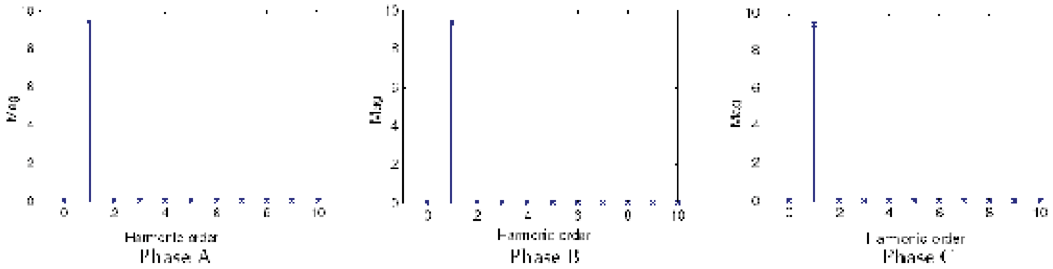


Figure 16. Harmonic spectrum of each phase of a healthy inverter (simulation results).

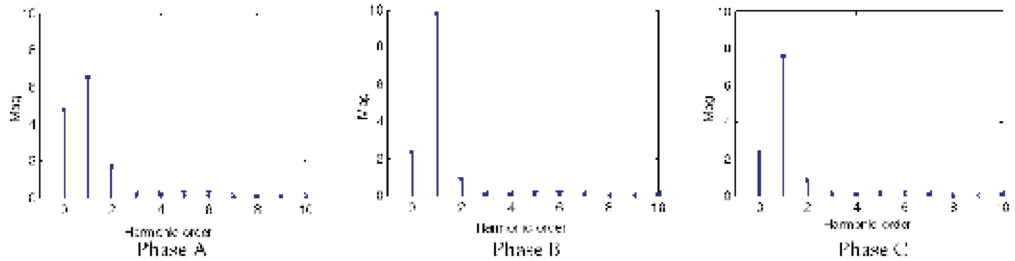


Figure 17. Harmonic spectrum of each phase of a faulty inverter (simulation results).

The analysis of the first eight harmonics shows that the difference between the healthy state and the case of open-circuit fault state lies at the zero-order harmonics which means the presence of the DC component in the signal. The argument of zero harmonic relative to the fundamental enables us to know the type of fault. On the other hand, the argument of this harmonic enables us also to know the faulty switch either the upper one or the lower one.

From the result of **Table 7**, we note that the phase which contains the open-circuit fault has its dc component equals to the sum of the dc component of the two other phases and is expressed by the following relations as:

- If the fault is at phase A, then: $h_{0A} = h_{0B} + h_{0C}$
- If the fault is at phase B, then: $h_{0B} = h_{0A} + h_{0C}$
- If the fault is at phase C, then: $h_{0C} = h_{0A} + h_{0B}$

Where h_{0A} is the zero-order harmonic of phase A, h_{0B} the zero-order harmonic of phase B and h_{0C} the zero-order harmonic of phase C.

5.2. Experimental results for healthy and faulty inverter using the stator current spectrum analysis

In what follows we will present the experimental results for the healthy and open-circuit faulty inverter. **Figure 18** shows the harmonic spectrum of the current for the healthy state, while **Figure 19** depicts the experimental results for open-circuit fault state.

	Zero-order harmonic of the three phases		
Faults types	**Phase A**	**Phase B**	**Phase C**
Healthy case	$\lvert h_0\rvert = \varepsilon_{h0}$, $(h_0 = 0)$	$\lvert h_0\rvert = \varepsilon_{h0}$, $(h_0 = 0)$	$\lvert h_0\rvert = \varepsilon_{h0}$, $(h_0 = 0)$
K_1 *open*	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 5.377)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 2.688)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 2.688)$
K_2 *open*	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0(h_0 = 5.35)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 2.68)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 2.68)$
K_3 *open*	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 2.68)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 5.36)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 2.68)$
K_4 *open*	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 2.69)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 5.39)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 2.69)$
K_5 *open*	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 2.69)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 2.69)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 5.38)$
K_6 *open*	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 2.69)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 270^0$, $(h_0 = 2.69)$	$\varepsilon_{h0} < \lvert h_0\rvert < \lvert h_1\rvert$ $\varphi_{h0} = 90^0$, $(h_0 = 5.37)$

Table 7. Open-circuit fault characteristics of an inverter.

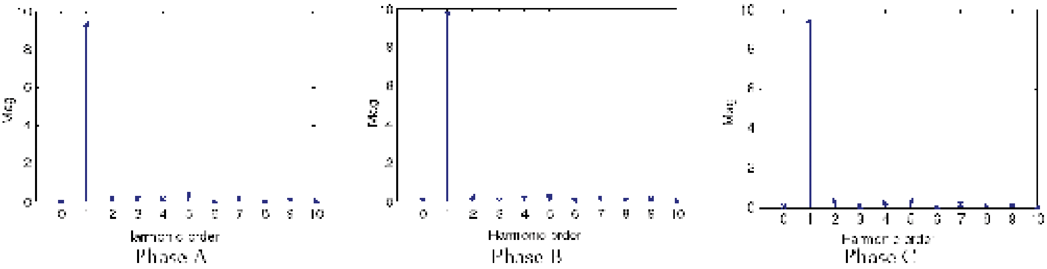


Figure 18. Harmonic spectrum of each phase of a healthy inverter (experimental results).

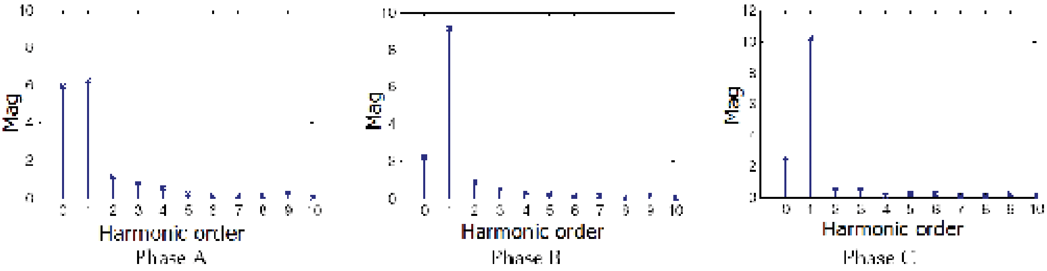


Figure 19. Harmonic spectrum of each phase in case of a faulty inverter (experimental results).

6. Technique based on the measure of the current drop

This technique requires the use of two current sensors by inverter leg. The current measurement may be performed using Hall Effect sensors. Hence the two current sensors for

C_{K1}	C_{K2}	$i_{K1} < -i_0$	$i_{K2} < -i_0$	Conclusion
1	*0*	*0*	*1*	K_1 open
0	*1*	*1*	*0*	K_2 open

Table 8. Fault detection of a faulty inverter using the measure of the current drop technique.

each leg give the measurements of currents $\boldsymbol{i_{K1}}$ and $\boldsymbol{i_{K2}}$. If the current of phase A is positive, the K_1 switch is under open-circuit fault if it is ordered to close $\boldsymbol{C_{K1} = 1}$ but the current $\boldsymbol{i_{K2}}$ remains negative. For the K_2 switch and when ik1 is negative, the open-circuit fault is detected if $\boldsymbol{C_{K2} = 1}$ and $\boldsymbol{i_{K1}}$, remain negative. We therefore obtain the following equations as given in [16]:

$$\boldsymbol{OC_{K1}} = \boldsymbol{C_{K1}}(\boldsymbol{i_{K2}}<-\boldsymbol{i_0}) \tag{12}$$

$$\boldsymbol{OC_{K2}} = \boldsymbol{C_{K2}}(\boldsymbol{i_{K1}}<-\boldsymbol{i_0}) \tag{13}$$

where $\boldsymbol{C_{K1} = 1}$ is the Control of K_1 IGBT and $\boldsymbol{C_{K2} = 1}$ is the Control of K_2 IGBT (**Table 8**).

The algorithm of the open-circuit detection applied in this technique; as shown in **Figure 20**; is based on the measurement of the current drop and can be described by the following steps:

1st step: Measure the current I_j.

2nd step: Search the current error e_j by comparing the current I_j to the threshold current I_0.

3th step: Identify whether any of the six errors exceed the threshold I_0.

4th step: If so, identify the faulty leg of the inverter that should be immediately isolated.

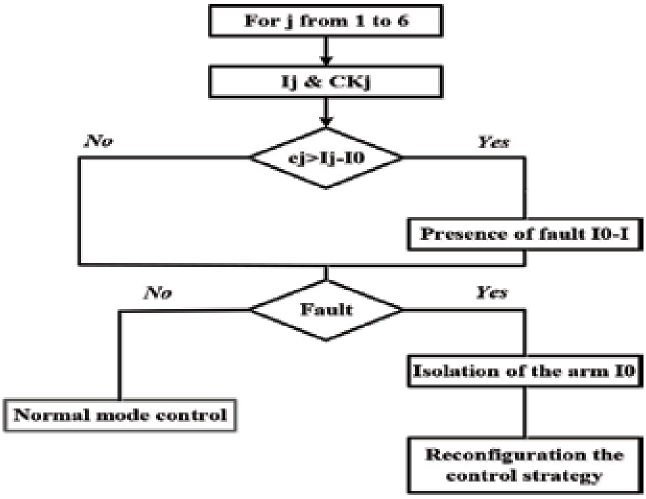


Figure 20. Algorithm for detecting an inverter open-circuit fault using the measure of the current drop.

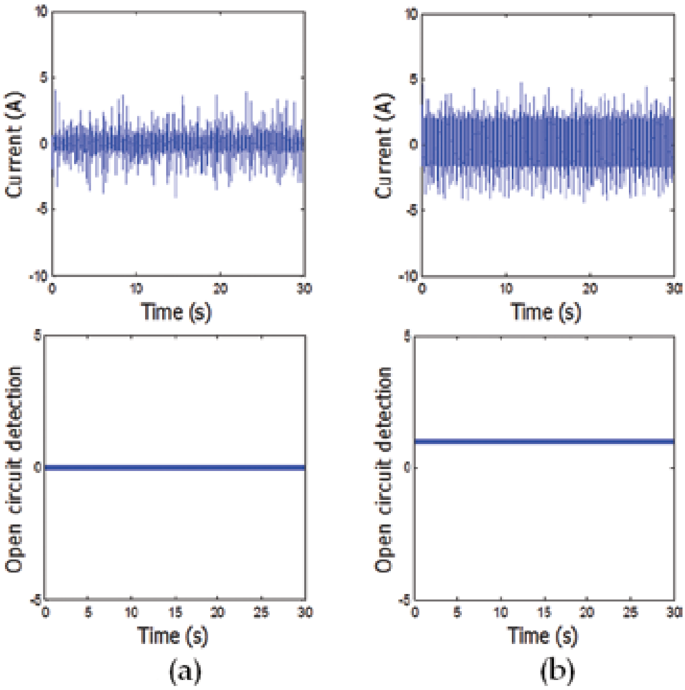


Figure 21. Healthy and faulty inverters detection (a) healthy inverter and (b) open-circuit fault.

In what follows, the experimental results for the case of a healthy inverter state and an open-circuit fault of the K_1 switch state, of an inverter are represented as shown in **Figure 21**, using the technique based on the measurement of the switch current drop.

7. Comparative study between the different techniques

The comparative study is carried out between different detection techniques. The aim of this comparison is to investigate and evaluate the performance of each detection technique studied. The comparison study focuses on the time detection, the localization ability, the hardware aspect and also the error between the threshold set to zero and the fault value. From this comparison study as illustrated both in **Figure 22** and **Table 9**, we come up with the following deductions: the fastest technique in terms of open-circuit fault time detection (*1.5 ms*) is the measurement of the switch current drop technique. Unfortunately this technique presents also two disadvantages: the first one is related to its inability for localization and is used only for detection purpose. The second drawback of this technique is concerned with the hardware and implementation aspect as it utilizes six current sensors (one for each gate of the six IGBT gates of the three phase two-level voltage source inverter).

The three other techniques: the one based on the Park vectors witch polar coordinate, the one based on the average value of the currents and the one based on the current spectral analysis

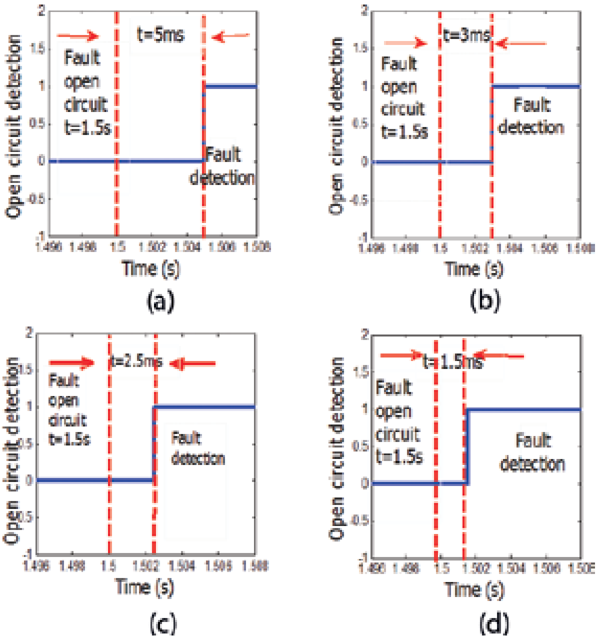


Figure 22. Detection time of open-circuit fault for the four techniques, (a) Park vectors with polar coordinates, (b) mean value of the currents, (c) current spectrum analysis and (d) measure of the current drop.

Techniques	Hardware	Detection time (*ms*)	Detection	Localization	Error	Quality and performance
Park stator current Vectors	Three current sensors	*5*	Yes	Yes	7.62	Good
Average value of stator currents	Three current sensors	*3*	Yes	Yes	1.5	Good
Stator current spectrum analysis	Three current sensors	**2.5**	Yes	Yes	5.37	Very good
Measure of the stator current drop	Six current sensors	*1.5*	Yes	No	3	Average

Table 9. Comparison summary of the detection techniques.

are all used for the detection and localization of open-circuit switch faults in inverters. It should be noted that the three techniques require only three sensors, hence presenting an advantage over the measurement of the current drop technique in terms of cost and implementation. The current spectral analysis has an additional advantage in terms of detection time rapidity (*2.5 ms*) compared to the two other ones (*3* and *5 ms*) respectively. This comparative study is summarized in **Table 9**. For all detection techniques presented in this chapter study, the calculated error in **Table 9**, is that between the normal value corresponding to the threshold for the healthy state case (note that the healthy state is taken as a reference corresponding to zero value of the threshold) relative to the fault value corresponding to the IGBT switch open-circuit fault state.

In what follows, **Figure 22**. illustrates the detection time of the four techniques used for open-circuit fault occurrence at *t = 1.5 ms*.

8. Conclusion

In this chapter work, the main purpose is to present a very detailed study of some detection techniques using both simulation and experimental work. The study is followed by a performance comparison between the various techniques to illustrate the feasibility and merits of each one of them and shows the suitability of each technique for a diagnostic situation. The study focuses mainly on the detection time as a key parameter in the detection procedure, the ability of the technique to localize in an exact manner the inverter IGBT switch open-circuit fault and also to assess the hardware aspect in terms of number of sensors required. The experimental work is conducted to validate the simulation results obtained.

The Technique based on the measure of the current drop is found to be superior in terms of fault time detection rapidity but unfortunately cannot be used for fault localization which is a drawback with respect to the three other presented techniques. Another disadvantage of this technique is the high number of required current sensors which has doubled in comparison to the three other studied techniques.

The three other techniques based on the Park vectors associated with polar coordinates, the average value of the currents and the current spectral analysis can be all used for detection and localization of inverter IGBT open-circuit faults and they require only three sensors, hence presenting an advantage over the measure of the current drop technique in terms of cost and implementation. The current spectrum analysis has an additional advantage in terms of detection time rapidity (*2.5 ms*) compared to the two other ones (*3 ms* and *5 ms*).

The chapter also proposes polar coordinates calculation to enhance the Park current vectors technic by developing a simple graphical representation which enables the exact computation of both the magnitude and the phase angle of the fault current vector related to the faulty inverter IGBT switch.

Author details

Bilal Djamal Eddine Cherif*, Azeddine Bendiabdellah, Mokhtar Bendjebbar and Laribi Souad

*Address all correspondence to: bilal.cherif@univ-usto.dz

Diagnostic Group, Laboratory LDEE, Electrical Engineering Faculty, University of Sciences
and Technology of Oran, Bir El Djir, Algeria

References

[1] Cherif BDE, Bendiabdellah A, Khelif MA. Detection of open-circuit fault in a three-phase voltage inverter fed induction motor. International Review of Automatic Control. 2016; **9**(6):374-382

[2] Eddine CBD, Azzeddine B, Amine KM, Mokhtar B, Noureddine B. The enhancement of park current vectors technique for inverter fault detection. In: Proceedings of the 6th International Conference on Systems and Control; University of Batna 2, Algeria; May 7–9, 2017. pp. 377-382

[3] Orlowska-Kowalska T, Sobanski P. Simple diagnostic technique of a single IGBT open-circuit faults for a SVM-VSI vector controlled induction motor drive. Bulletin of the Polish Academy of Sciences—Technical Sciences. 2015;**63**(1):281-288

[4] Cherif BDE, Bendjebbar M, Benouzza N, Boudinar H, Bendiabdellah A. A comparative study between two open-circuit fault detection and localization techniques in a three phase inverter fed induction motor. In: Proceedings of the 2016 8th International Conference on Modelling, Identification and Control (ICMIC); Algiers; November 2016. pp. 1-7

[5] Zhang W, Xu D, Enjeti PN, Li H, Hawke JT, Krishnamoorthy HS. Survey on fault-tolerant techniques for power electronic converters. IEEE Transactions on Power Electronics. 2014; **29**:6319-6331

[6] Orlowska-Kowalska T, Sobanski P. Simple sensorless diagnosis method for open-switch faults in SVM-VSI-fed induction motor drive. In: IEEE 39th Ann. Conf. of Ind. Electron. Soc.; 2013. pp. 8210-8215

[7] Asghar F, Talha M, Kim SH. Comparative study of three fault diagnostic methods for three phase inverter with induction motor. International Journal of Fuzzy Logic and Intelligent Systems;**17**(4):245-256

[8] Jlassi I, Khojet S, Khil E. A MRAS-Luenberger observer based fault tolerant control of PMSM drive. Journal of Electrical Systems. 2014;**10**(1):48-62

[9] Cherif BDE, Bendjebbar M, Bendiabdellah A. Diagnosis of open-circuit fault in a three phase voltage inverter fed induction motor. In: The 4th International Conference on Electrical Engineering – ICEE'2015; Boumerdes, Algeria

[10] Estima JO, Freire NMA, Cardoso AJM. Recent advances in fault diagnosis by Park's vector approach. IEEE workshop Electr. Mach. Des Control and Diagn. vol. 2. 2013. pp. 279-288

[11] Mendes AMS, Cardoso AJM. Fault diagnosis in a rectifier-inverter system used in variable speed ac drives by the average current Park's vector approach. In: 8th European conference on power electronics and applications, EPE'99. 1999. pp. 1-9

[12] Abramik S, Sleszynski W, Nieznanski J, Piquet HA. Diagnostic method for on-line fault detection and localization in VSI-fed ac drives. In: 10th European Conference on Power Electronics and Applications; Toulouse, France. 2003

[13] Rothenhagen K, Fuchs FW. Performance of diagnosis methods for IGBT open circuit faults in voltage source active rectifiers. In: Proceedings of the .IEEE PESC; 2004. pp. 4348-4354

[14] Cherif BDE, Bendiabdellah A. Detection of two-level inverter open circuit fault using a combined DWT-NN approach, Hindawi. Journal of Control Science and Engineering. Vol. 2018. 11 Pages. Article ID: 1976836

[15] Sharma SK, Lu B. Literature review of IGBT fault diagnostic and protection methods for power inverters. IEEE Transactions on Industry Applications. 2009;**45**(5):1770-1777

[16] Raison B, Rostaing G, Rognon JP. Towards a global monitoring scheme for induction motor drives. In: Int. Power Electron. Conf; Tokyo. 2000. pp. 1183-1188

Section 4

Kalman Filtering

Chapter 5

Fault Residuals Based on Distributed Discrete-Time Linear Kalman Filtering

Dušan Krokavec and Anna Filasová

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.80296

Abstract

The chapter is concerned with the application of distributed discrete-time linear Kalman filtering with decentralized structure of sensors in fault residual generation. Two variants of distributed Kalman filtering algorithms are introduced, giving the incidence of equivalent functional realization structure of fault residual filters. The obtained solutions use Kalman filter innovations in a nonstandard way to generate residuals with significantly higher dynamic signal range. The obtained results, offering structures for fault detection filter realization, are illustrated with a numerical example to note the effectiveness of the approach.

Keywords: linear noisy systems, Kalman filtering, innovation sequences, fault residual filters, distributed computing

1. Introduction

The castigatory principal aspect for designing a fault-tolerant control (FTC) structure is a functionality of diagnostic operations that solve the fault detection and isolation (FDI) tasks. These techniques most commonly use residuals generated by fault detection filters (FDF), followed by the residual signal evaluation within decision functions. Guarantying adequate sensitivity to faults, the accessory objective is to create residuals with minimal sensitivity to noises. Kalman filtering is an optimal state estimation process applied to a dynamic system that involves random noises, giving a linear, unbiased, and minimum error variance recursive algorithm to optimally estimate the unknown state of a dynamic system from noisy data taken from sensors [1, 2].

IntechOpen

Practically, a bank of Kalman filters is used to achieve sensor and actuator fault detection applied to a steady-state system, while the statistical characteristics of the system are not required to be known after a fault has occurred [3, 4]. In these methods, the faults are assumed to be known, and the Kalman filters are designed for such kind of sensor or actuator faults. Another approach based on Kalman filtering is the analysis of the innovation sequence, since faults displace its zero mean and change its covariance matrix [5]. The associated problem is quick detection of changes in these parameters from their nominal values. Evidently, research in Kalman filter based-FDI is the subject of wide range of other publications (see, e.g., [6–9] and the reference therein). Other applications can be found in [10].

The state estimation obtained by the Kalman filter prediction-correction equations at every time instant can be solved almost optimally and substantially faster by applying a distributed approach [11–14]. With this setup can be exploited the fact that the correction error can be decaying exponentially with time instant sequence to reach the optimal values [15–18].

The chapter exploits a variant of distributed methods to apply the distributed correction stage filtering equations on each sensor level as well as an approach based on quasi-parallel central computation. Benefiting from the distributed Kalman filtering algorithm, two residually equivalent signal structures are presented for the discrete-time linear noisy systems.

The outline of this chapter is as follows: Section 1 delineates the problem and draws the basic starting points of solutions. Dealing with the discrete-time noisy systems description, the equations describing Kalman filters for uncorrelated measurement and system noises are traced out in Section 2, to delineate distributed approaches in Kalman filter design, suitable for supporting the fault residual generation, presented in Section 3. Section 4 gives a numerical example, illustrating the properties of the proposed method, and Section 5 presents some concluding remarks.

Throughout the chapter, the notations are narrowly standard in such a way that $\boldsymbol{x}^T$ and $\boldsymbol{X}^T$ denote the transpose of vector $\boldsymbol{x}$ and matrix $\boldsymbol{X}$, respectively, and $diag[\cdot]$ denotes a block diagonal matrix—for a square matrix $\boldsymbol{X} > 0$ means that $\boldsymbol{X}$ is a symmetric positive definite matrix. The symbol $\boldsymbol{I}_n$ indicates the nth order unit matrix; $\mathbb{R}$ denotes the set of real numbers; $\mathbb{R}^n$ and $\mathbb{R}^{n \times r}$ refer to the set of all n-dimensional real vectors and $n \times r$ real matrices, respectively; and $\mathcal{Z}_+$ is the set of all positive integers.

2. Discrete-time linear Kalman filter

In this section, one version of the Kalman filtering concept is applied for the discrete-time linear multi inputs and multi outputs (MIMO) plants with the system and output noises of the form

$$\boldsymbol{q}(i+1) = \boldsymbol{F}\boldsymbol{q}(i) + \boldsymbol{G}\boldsymbol{u}(i) + \boldsymbol{v}(i), \tag{1}$$

$$\boldsymbol{y}(i) = \boldsymbol{C}\boldsymbol{q}(i) + \boldsymbol{o}(i), \tag{2}$$

where $\boldsymbol{q}(i) \in \mathbb{R}^n$, $\boldsymbol{u}(i) \in \mathbb{R}^r$, and $\boldsymbol{y}(i) \in \mathbb{R}^m$ are vectors of the system state, input and measurement output variables, respectively; $\boldsymbol{v}(i) \in \mathbb{R}^n$ and $\boldsymbol{o}(i) \in \mathbb{R}^m$ are vectors of the system and measurement noise; and $\boldsymbol{F} \in \mathbb{R}^{n \times n}$, $\boldsymbol{G} \in \mathbb{R}^{n \times r}$, and $\boldsymbol{C} \in \mathbb{R}^{m \times n}$ are conditioned by $1 \le m, r \le n$. Kalman filter is used only for diagnostic purposes. Zero-mean Gaussian white noise processes are considered such that

$$E\left\{\begin{bmatrix} \boldsymbol{v}(i) \\ \boldsymbol{o}(i) \end{bmatrix}\right\} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \tag{3}$$

$$E\left\{\begin{bmatrix} \boldsymbol{v}(i) \\ \boldsymbol{o}(i) \end{bmatrix}\begin{bmatrix} \boldsymbol{v}^T(k) & \boldsymbol{o}^T(k) \end{bmatrix}\right\} = \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{S} \\ \boldsymbol{S}^T & \boldsymbol{R} \end{bmatrix}\delta_{ik}, \tag{4}$$

where $E\{\cdot\}$ is the a statistical averaging operator,

$$\delta_{ik} = \begin{cases} 1 & \text{if } i = k, \\ 0 & \text{if } i \ne k, \end{cases} \tag{5}$$

is the Kronecker delta-function and the covariance matrices $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{R} \in \mathbb{R}^{m \times m}$ are symmetric positive definite matrices.

It is assumed that the deterministic system initial state $\boldsymbol{q}(0) = \boldsymbol{q}_0$ is independent of $\boldsymbol{v}(i)$ and $\boldsymbol{o}(i)$ in the sense that

$$E\{\boldsymbol{q}_0\boldsymbol{v}^T(i)\} = \mathbf{0}, \quad E\{\boldsymbol{q}_0\boldsymbol{o}^T(i)\} = \mathbf{0} \quad \text{for all } i \tag{6}$$

and that the system and measurement noises are uncorrelated, i.e., $\boldsymbol{S} = \mathbf{0}$.

Determining the optimal system state vector estimate, $\boldsymbol{q}_e(i|i-1)$ denotes the predicted estimation of the system state vector $\boldsymbol{q}(i)$ at the time instant i in the dependency on all noisy output measurement vector sequence $\{\boldsymbol{y}(j),\ j = 0, 1, \dots, i-1\}$ up to time instant $i-1$; $\boldsymbol{q}_e(i|i)$ is the corrected estimation of the system state vector $\boldsymbol{q}(i)$ at the time instant i in the dependency on all noisy output measurement sequence $\{\boldsymbol{y}(j),\ j = 0, 1, \dots, i\}$ up to time instant i; and $\boldsymbol{e}(i|i-1) = \boldsymbol{q}(i) - \boldsymbol{q}_e(i|i-1)$ and $\boldsymbol{e}(i|i) = \boldsymbol{q}(i) - \boldsymbol{q}_e(i|i)$ are prediction and correction errors.

Definition 1. *[19] If the Kalman filter, associated with the plant (1), (2) with uncorrelated system and measurement noises, is defined by the set of equations*

$$\boldsymbol{q}_e(i|i-1) = \boldsymbol{F}\boldsymbol{q}_e(i-1|i-1) + \boldsymbol{G}\boldsymbol{u}(i-1), \tag{7}$$

$$\boldsymbol{q}_e(i|i) = \boldsymbol{q}_e(i|i-1) + \boldsymbol{J}(i)\big(\boldsymbol{y}(i) - \boldsymbol{y}_e(i|i-1)\big), \tag{8}$$

$$\boldsymbol{y}_e(i|i-1) = \boldsymbol{C}\boldsymbol{q}_e(i|i-1), \tag{9}$$

$$\boldsymbol{y}_e(i|i) = \boldsymbol{C}\boldsymbol{q}_e(i|i), \tag{10}$$

then with $\boldsymbol{q}_e(0|0) = \boldsymbol{q}_0$, $\boldsymbol{P}(0|0) = \boldsymbol{Q}^\circ$, where $\boldsymbol{Q}^\circ \in \mathbb{R}^{n \times n}$ is a positive definite matrix, yielding

$$\boldsymbol{J}(i) = \boldsymbol{P}(i|i-1)\boldsymbol{C}^T\big(\boldsymbol{R} + \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T\big)^{-1}, \tag{11}$$

$$\boldsymbol{P}(i|i-1) = \boldsymbol{F}\boldsymbol{P}(i-1|i-1)\boldsymbol{F}^T + \boldsymbol{Q}, \tag{12}$$

$$\boldsymbol{P}(i|i) = (\boldsymbol{I} - \boldsymbol{J}(i)\boldsymbol{C})\boldsymbol{P}(i|i-1), \tag{13}$$

where

$$\boldsymbol{P}(i|i-1) = E\{\boldsymbol{e}(i|i-1)\boldsymbol{e}^T(i|i-1)\}, \tag{14}$$

$$\boldsymbol{P}(i|i) = E\{\boldsymbol{e}(i|i)\boldsymbol{e}^T(i|i)\}, \tag{15}$$

are the covariance matrices of prediction and correction errors and $\boldsymbol{J}(i) \in \mathbb{R}^{n\times m}$ is the Kalman filter gain matrix, all at time instant i.

The discrete-time Kalman filter equations can be algebraically manipulated into a variety of forms [6, 16, 20]. From the point of view of distributed filtration, it is necessary to achieve such form of the equation for calculating the Kalman gain $\boldsymbol{J}(i)$ that yields the matrix $\boldsymbol{C}$ from the matrix inversion operation (see (11)). If the system and measurement noises are uncorrelated, then for the Kalman filter gain, one can propose the following:

Lemma 1. *If the system and measurement noises are uncorrelated, then the Kalman filter gain and the correction error covariance matrix can be computed using (12) and*

$$\boldsymbol{J}(i) = \boldsymbol{P}(i|i)\boldsymbol{C}^T\boldsymbol{R}^{-1}, \tag{16}$$

$$\boldsymbol{P}^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}. \tag{17}$$

Proof. Substituting (11) into (13), one can obtain that

$$\boldsymbol{P}(i|i) = \boldsymbol{P}(i|i-1) - \boldsymbol{P}(i|i-1)\boldsymbol{C}^T\left(\boldsymbol{R} + \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T\right)^{-1}\boldsymbol{C}\boldsymbol{P}(i|i-1). \tag{18}$$

Exploiting the Sherman-Morrison-Woodbury formula of the form [21].

$$\left(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{D}\boldsymbol{B}^T\right)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}\left(\boldsymbol{D}^{-1} + \boldsymbol{B}^T\boldsymbol{A}^{-1}\boldsymbol{B}\right)^{-1}\boldsymbol{B}^T\boldsymbol{A}^{-1}, \tag{19}$$

where square invertible matrices $\boldsymbol{A}$, $\boldsymbol{D}$, and a matrix $\boldsymbol{B}$ of appropriate dimensions are such that $\left(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{D}\boldsymbol{B}^T\right)$ is invertible, with

$$\boldsymbol{A} = \boldsymbol{P}(i|i-1), \quad \boldsymbol{B} = \boldsymbol{P}(i|i-1)\boldsymbol{C}^T, \quad \boldsymbol{D} = -\left(\boldsymbol{R} + \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T\right)^{-1}, \tag{20}$$

yields, since the covariance matrices are positive definite,

$$\boldsymbol{P}^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) - \boldsymbol{P}^{-1}(i|i-1)\boldsymbol{P}(i|i-1)\boldsymbol{C}^T\boldsymbol{E}^{-1}\boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{P}^{-1}(i|i-1), \tag{21}$$

where

$$\boldsymbol{E} = -\boldsymbol{R} - \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T + \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{P}^{-1}(i|i-1)\boldsymbol{P}(i|i-1)\boldsymbol{C}^T = -\boldsymbol{R}. \tag{22}$$

Then, evidently,

$$\boldsymbol{P}^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) - \boldsymbol{C}^T\boldsymbol{E}^{-1}\boldsymbol{C} \tag{23}$$

and (31) implies (17).

Premultiplying the left side by $\boldsymbol{P}^{-1}(i|i)$ and postmultiplying the right side by $\boldsymbol{P}^{-1}(i|i-1)$, (13) gives

$$\boldsymbol{P}^{-1}(i|i-1) = \boldsymbol{P}^{-1}(i|i) - \boldsymbol{P}^{-1}(i|i)\boldsymbol{J}(i)\boldsymbol{C} \tag{24}$$

and comparing (17) and (24), it can be seen that

$$\boldsymbol{P}^{-1}(i|i)\boldsymbol{J}(i)\boldsymbol{C} = \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}. \tag{25}$$

Thus, (25) implies (16). This concludes the proof.

Note, since $\boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}$ is at least a positive semi-definite matrix, it is evident from (17) that $\boldsymbol{P}(i|i)$ is never larger than $\boldsymbol{P}(i|i-1)$. Moreover, the result is an unbiased filter with the estimates of minimum error variances. More details can be found in [12, 22].

Corollary 1. *Considering that $\boldsymbol{q}_e(i|i-1)$ is known and $\boldsymbol{q}_e(i|i)$ is the best estimate of $\boldsymbol{q}(i)$ that minimizes the cost criterion*

$$\begin{aligned} T(i) = &\left(\boldsymbol{q}(i) - \boldsymbol{q}_e(i|i-1)\right)^T\boldsymbol{P}^{-1}(i|i-1)\left(\boldsymbol{q}(i) - \boldsymbol{q}_e(i|i-1)\right) \\ &+ (\boldsymbol{y}(i) - \boldsymbol{C}\boldsymbol{q}(i))^T\boldsymbol{R}^{-1}(\boldsymbol{y}(i) - \boldsymbol{C}\boldsymbol{q}(i)). \end{aligned} \tag{26}$$

Then, evaluating (26) it follows, with the optimal setting of a state vector estimate $\boldsymbol{q}(i) = \boldsymbol{q}(i|i)$, that the minimum expected cost is given by

$$\frac{dT(i)}{d\boldsymbol{q}(i)^T} = \boldsymbol{P}^{-1}(i|i-1)\left(\boldsymbol{q}(i|i) - \boldsymbol{q}_e(i|i-1)\right) - \boldsymbol{C}^T\boldsymbol{R}^{-1}(\boldsymbol{y}(i) - \boldsymbol{C}\boldsymbol{q}(i|i)) = 0, \tag{27}$$

which implies

$$\begin{aligned} \left(\boldsymbol{P}^{-1}(i|i-1) + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}\right)\boldsymbol{q}_e(i|i) &= \boldsymbol{P}^{-1}(i|i-1)\boldsymbol{q}_e(i|i-1) + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{y}(i) \\ &= \left(\boldsymbol{P}^{-1}(i|i-1) + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}\right)\boldsymbol{q}_e(i|i-1) \\ &\quad + \boldsymbol{C}^T\boldsymbol{R}^{-1}\left(\boldsymbol{y}(i) - \boldsymbol{C}\boldsymbol{q}_e(i|i-1)\right). \end{aligned} \tag{28}$$

Therefore, using the above relations, at the ith step Eq. (28) gives

$$\begin{aligned} \boldsymbol{q}_e(i|i) &= \boldsymbol{q}_e(i|i-1) + \left(\boldsymbol{P}^{-1}(i|i-1) + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}\right)^{-1}\boldsymbol{C}^T\boldsymbol{R}^{-1}\left(\boldsymbol{y}(i) - \boldsymbol{C}\boldsymbol{q}_e(i|i-1)\right) \\ &= \boldsymbol{q}_e(i|i-1) + \boldsymbol{P}(i|i) \times \boldsymbol{C}^T\boldsymbol{R}^{-1}\left(\boldsymbol{y}(i) - \boldsymbol{C}\boldsymbol{q}_e(i|i-1)\right). \end{aligned} \tag{29}$$

Pre-multiplying the left side by $\boldsymbol{P}(i|i)$ and post-multiplying the right side by $\boldsymbol{P}(i|i-1)$ then it follows from (17)

$$\boldsymbol{P}(i|i-1) = \boldsymbol{P}(i|i) + \boldsymbol{P}(i|i)\boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}\boldsymbol{P}(i|i-1), \tag{30}$$

which can be proved recursively as follows

$$\boldsymbol{P}(i|i) = \left(\boldsymbol{I}_n - \boldsymbol{P}(i|i)\boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C}\right)\boldsymbol{P}(i|i-1). \tag{31}$$

Comparing (29) with the covariance matrix of the filtering error given by (13), it is evident that

$$\boldsymbol{J}(i) = \boldsymbol{P}(i|i)\boldsymbol{C}^T\boldsymbol{R}^{-1} \tag{32}$$

which is identical to (16).

On the other side, substituting (11) into (13), one can write

$$\boldsymbol{P}(i|i) = \boldsymbol{P}(i|i-1) - \boldsymbol{P}(i|i-1)\boldsymbol{C}^T\left(\boldsymbol{R} + \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T\right)^{-1}\boldsymbol{C}\boldsymbol{P}(i|i-1) \tag{33}$$

and using the Sherman-Morrison-Woodbury formula, Eq. (27), it follows

$$\boldsymbol{P}^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) - \boldsymbol{C}^T\left(-\boldsymbol{R} - \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T + \boldsymbol{C}\boldsymbol{P}(i|i-1)\boldsymbol{C}^T\right)^{-1}\boldsymbol{C} \tag{34}$$

and so, evidently, (34) gives (17).

3. Fault residual generation using distributed Kalman filtering

The obtained equations, Eqs. (16) and (17), allow the use of the open form of the Kalman filter equations if

$$\boldsymbol{R} = E\left\{\boldsymbol{o}(i)\boldsymbol{o}^T(i)\right\} = \mathrm{diag}[\,R_1 \quad R_2 \quad \cdots \quad R_m\,]. \tag{35}$$

Writing separately,

$$\mathbf{y}^T(i) = \left[\,y_1(i) \quad y_2(i) \quad \cdots \quad y_m(i)\,\right], \tag{36}$$

$$\boldsymbol{u}^T(i) = [\,u_1(i) \quad u_2(i) \quad \cdots \quad u_r(i)\,], \tag{37}$$

$$\boldsymbol{C}^T = [\,\boldsymbol{c}_1 \quad \boldsymbol{c}_2 \quad \ldots \quad \boldsymbol{c}_m\,], \quad \boldsymbol{G} = \left[\,\boldsymbol{g}_1 \quad \boldsymbol{g}_2 \quad \cdots \quad \boldsymbol{g}_r\,\right], \tag{38}$$

then, (7)–(11), (16), and (27) imply

$$\boldsymbol{q}_e(i|i-1) = \boldsymbol{F}\boldsymbol{q}_e(i-1|i-1) + \sum_{h=1}^{w}\boldsymbol{g}_h u_h(i-1), \tag{39}$$

$$\boldsymbol{q}_e(i|i) = \boldsymbol{q}_e(i|i-1) + \sum_{h=1}^{m}\boldsymbol{j}_h(i)\left(\boldsymbol{y}_h(i) - \boldsymbol{c}_h^T\boldsymbol{q}_e(i|i-1)\right), \tag{40}$$

$$y_{ej}(i|i-1) = \boldsymbol{c}_h^T\boldsymbol{q}_e(i|i-1), \tag{41}$$

$$y_{ej}(i|i) = \boldsymbol{c}_h^T \boldsymbol{q}_e(i|i), \tag{42}$$

$$\boldsymbol{j}_h(i) = \boldsymbol{P}(i|i)\boldsymbol{c}_h R_h^{-1}, \tag{43}$$

$$\boldsymbol{P}^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) + \sum_{h=1}^{m} \boldsymbol{c}_h R_h^{-1} \boldsymbol{c}_h^T, \tag{44}$$

$$\boldsymbol{P}(i|i-1) = \boldsymbol{F}\boldsymbol{P}(i-1|i-1)\boldsymbol{F}^T + \boldsymbol{Q}. \tag{45}$$

It is evident from the above given formulation that the relation of (40) gives the possibility to compute corrections from the data obtained at all sensor nodes.

Theorem 1. *Defining the residual vector as*

$$\boldsymbol{r}^T(i) = [\, z_1(i) \quad z_2(i) \quad \dots \quad z_m(i) \,], \tag{46}$$

where

$$z_h(i) = \boldsymbol{y}_h(i) - \boldsymbol{c}_h^T \boldsymbol{q}_{ec}(i|i-1), \tag{47}$$

then

$$\boldsymbol{q}_{ec}(i|i-1) = \boldsymbol{F}\boldsymbol{q}_{ec}(i-1|i-1) + \sum_{h=1}^{r} \boldsymbol{g}_h u_h(i-1), \tag{48}$$

$$\boldsymbol{q}_{ed}(i|i-1) = \boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1), \tag{49}$$

$$\boldsymbol{q}_{ec}(i|i) = \boldsymbol{q}_{ec}(i|i-1), \tag{50}$$

$$\boldsymbol{q}_{ed}(i|i) = \boldsymbol{q}_{ed}(i|i-1) + \boldsymbol{P}(i|i) \sum_{h=1}^{m} \boldsymbol{c}_h R_h^{-1} \big(z_h(i) - \boldsymbol{c}_h^T \boldsymbol{q}_{ed}(i|i-1) \big), \tag{51}$$

while the filter gain matrices, as well as recurrences of the covariance matrices are given by (43)–(45).

Proof. Considering that there are components of the system state vector estimate that are dependent on the control signal as well as ones that are not dependent on the control signals, since the correction step does not depend on the control inputs, (40) can be rewritten as

$$\boldsymbol{q}_e(i|i) = \boldsymbol{q}_{ec}(i|i-1) + \boldsymbol{q}_{ed}(i|i-1) + \sum_{h=1}^{m} \boldsymbol{j}_h(i) \big(\boldsymbol{y}_h(i) - \boldsymbol{c}_h^T \left(\boldsymbol{q}_{ec}(i|i-1) + \boldsymbol{q}_{ed}(i|i-1) \right) \big). \tag{52}$$

Prescribing that

$$\boldsymbol{q}_e(i|i) = \boldsymbol{q}_{ed}(i|i) + \boldsymbol{q}_{ec}(i|i), \tag{53}$$

Eqs. (52) and (53) can be separated as

$$\boldsymbol{q}_{ec}(i|i) = \boldsymbol{q}_{ec}(i|i-1), \tag{54}$$

$$\boldsymbol{q}_{ed}(i|i) = \boldsymbol{q}_{ed}(i|i-1) + \sum_{h=1}^{m} \boldsymbol{j}_h(i)\big(\boldsymbol{y}_h(i) - \boldsymbol{c}_h^T\big(\boldsymbol{q}_{ec}(i|i-1) + \boldsymbol{q}_{ed}(i|i-1)\big)\big) \tag{55}$$

and using (47), (54) gives (50), and (55) implies (51).

Substituting (53) in (39) yields

$$\boldsymbol{q}_e(i|i-1) = \boldsymbol{F}\big(\boldsymbol{q}_{ed}(i-1|i-1) + \boldsymbol{q}_{ec}(i-1|i-1)\big) + \sum_{h=1}^{w} \boldsymbol{g}_h u_h(i-1) \tag{56}$$

and, evidently, (56) implies (48) and (49). This concludes the proof. □

Remark 1. *If Eqs. (46)–(51) are analyzed from a computational point of view, it is clear that their structures support autonomous parallel calculations only with a single interaction defined by Eq. (47). However, the cost for this parallelism is additional computation at each step, but the directional properties of the components of the residual vector are advantageous in the case of single sensor faults. The directional sensor residual property derives indirectly from relationship (44). Since every component $z_h(i)$ carries with it the measurement noise $o_h(i)$ if $\boldsymbol{q}_{ed}(i|i-1)$ is used for LQG control, it will be no noise at the state control law input.*

In principle, it is possible to define the residue generation by results of the local system state correction at Kalman filtration at the time instant *i*.

Theorem 2. *Defining the residual vector as*

$$\boldsymbol{r}^T(i) = [\, z_1(i) \quad z_2(i) \quad \dots \quad z_m(i)\,], \tag{57}$$

where

$$z_h(i) = \boldsymbol{y}_h(i) - \boldsymbol{c}_h^T \boldsymbol{q}_{ec}(i|i-1), \tag{58}$$

then

$$\boldsymbol{q}_{edj}(i|i-1) = \boldsymbol{F}\boldsymbol{q}_{edj}(i-1|i-1), \tag{59}$$

$$\boldsymbol{q}_{edj}(i|i) = \boldsymbol{q}_{edj}(i|i-1) + \boldsymbol{j}_h(i)\Big(z_h(i) - \boldsymbol{c}_h^T \boldsymbol{q}_{edj}(i|i-1)\Big), \tag{60}$$

$$\boldsymbol{j}_h(i) = \boldsymbol{P}_h(i|i)\boldsymbol{c}_h R_h^{-1}, \tag{61}$$

$$\boldsymbol{P}_h^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) + \boldsymbol{c}_h R_h^{-1} \boldsymbol{c}_h^T, \tag{62}$$

while the predicted system state at the time instant i is computed centrally and the filtered full system state is covered by the equations

$$\begin{aligned}\boldsymbol{q}_{ed}(i|i) = & \sum_{h=1}^{m}\boldsymbol{P}(i|i)\boldsymbol{P}_h^{-1}(i|i)\boldsymbol{q}_{edj}(i|i)\\ & -\sum_{h=1}^{m}\boldsymbol{P}(i|i)\boldsymbol{P}^{-1}(i|i-1)\boldsymbol{F}\boldsymbol{q}_{edj}(i-1)|i-1))\\ & +\boldsymbol{P}(i|i)\boldsymbol{P}^{-1}(i|i-1)\boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1),\end{aligned} \tag{63}$$

$$\boldsymbol{q}_{ec}(i|i) = \boldsymbol{q}_{ec}(i|i-1) \tag{64}$$

at the time instants $i \in \mathcal{Z}_+$.

Proof. The correction step for the Kalman filter in Eq. (51) can be prescribed locally for the *j*th node as

$$\boldsymbol{q}_{edj}(i|i) = \boldsymbol{q}_{edj}(i|i-1) + \boldsymbol{j}_h(i)(\boldsymbol{z}_h(i) - \boldsymbol{z}_{dh}(i|i-1)), \tag{65}$$

where

$$\boldsymbol{z}_h(i) = \boldsymbol{y}_h(i) - \boldsymbol{c}_h^T\boldsymbol{q}_{ec}(i|i-1), \tag{66}$$

$$\boldsymbol{z}_{dh}(i|i-1) = \boldsymbol{c}_h^T\boldsymbol{q}_{edj}(i|i-1), \tag{67}$$

$$\boldsymbol{j}_h(i) = \boldsymbol{P}_h(i|i)\boldsymbol{c}_h R_h^{-1}, \tag{68}$$

$$\boldsymbol{P}_h^{-1}(i|i) = \boldsymbol{P}^{-1}(i|i-1) + \boldsymbol{c}_h R_h^{-1}\boldsymbol{c}_h^T. \tag{69}$$

Substituting (66), rearranging and postmultiplying the left side by $\boldsymbol{P}_h^{-1}(i|i)$, (65) implies

$$\boldsymbol{P}_h^{-1}(i|i)\Big(\boldsymbol{q}_{edj}(i|i) - \boldsymbol{q}_{edj}(i|i-1)\Big) = \boldsymbol{c}_h R_h^{-1}(\boldsymbol{z}_h(i) - \boldsymbol{z}_{dh}(i|i-1)), \tag{70}$$

$$\boldsymbol{c}_h R_h^{-1}\boldsymbol{z}_h(i) = \boldsymbol{c}_h R_h^{-1}\boldsymbol{c}_h^T\boldsymbol{q}_{edj}(i|i-1) + \boldsymbol{P}_h^{-1}(i|i)\Big(\boldsymbol{q}_{edj}(i|i) - \boldsymbol{q}_{edj}(i|i-1)\Big), \tag{71}$$

respectively. Since (69) gives

$$\boldsymbol{c}_h R_h^{-1}\boldsymbol{c}_h^T = \boldsymbol{P}_h^{-1}(i|i) - \boldsymbol{P}^{-1}(i|i-1), \tag{72}$$

with a simple elimination after inserting (72), (71) gives

$$\begin{aligned}\boldsymbol{c}_h R_h^{-1}\boldsymbol{z}_h(i) &= \boldsymbol{P}_h^{-1}(i|i)\Big(\boldsymbol{q}_{edj}(i|i) - \boldsymbol{q}_{edj}(i|i-1)\Big)\\ &\quad + \boldsymbol{P}_h^{-1}(i|i)\boldsymbol{q}_{edj}(i|i-1) - \boldsymbol{P}^{-1}(i|i-1)\boldsymbol{q}_{edj}(i|i-1)\\ &= \boldsymbol{P}_h^{-1}(i|i)\boldsymbol{q}_{edj}(i|i) - \boldsymbol{P}^{-1}(i|i-1)\boldsymbol{q}_{edj}(i|i-1).\end{aligned} \tag{73}$$

Combining (49) and (51) results in

$$\boldsymbol{q}_{ed}(i|i) = \boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1) + \boldsymbol{P}(i|i)\sum_{h=1}^{m}\boldsymbol{c}_h R_h^{-1}\big(\boldsymbol{z}_h(i) - \boldsymbol{c}_h^T\boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1), \tag{74}$$

which can be written as

$$\boldsymbol{q}_{ed}(i|i) = \sum_{h=1}^{m} \boldsymbol{P}(i|i)\boldsymbol{c}_h R_h^{-1} \boldsymbol{z}_h(i) + \left(\boldsymbol{I}_n - \sum_{h=1}^{n} \boldsymbol{P}(i|i)\boldsymbol{c}_h R_h^{-1} \boldsymbol{c}_h^T \right) \boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1). \tag{75}$$

Pre-multiplying the left side of (44) by $\boldsymbol{P}(i|i)$ leads to

$$\boldsymbol{I}_n - \sum_{h=1}^{w} \boldsymbol{P}(i|i)\boldsymbol{c}_h R_h^{-1} \boldsymbol{c}_h^T = \boldsymbol{P}(i|i)\boldsymbol{P}^{-1}(i|i-1) \tag{76}$$

and considering (76), relation (75) takes the form

$$\boldsymbol{q}_{ed}(i|i) = \sum_{h=1}^{m} \boldsymbol{P}(i|i)\boldsymbol{c}_h R_h^{-1} \boldsymbol{z}_h(i) + \boldsymbol{P}(i|i)\boldsymbol{P}^{-1}(i|i-1)\boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1). \tag{77}$$

Thus, the substitution of (73) into (77) gives

$$\begin{aligned} \boldsymbol{q}_{ed}(i|i) = {} & \sum_{h=1}^{m} \boldsymbol{P}(i|i)\boldsymbol{P}_h^{-1}(i|i)\boldsymbol{q}_{edj}(i|i) - \sum_{h=1}^{m} \boldsymbol{P}(i|i)\boldsymbol{P}^{-1}(i|i-1)\boldsymbol{q}_{edj}(i|i-1) \\ & + \boldsymbol{P}(i|i)\boldsymbol{P}^{-1}(i|i-1)\boldsymbol{F}\boldsymbol{q}_{ed}(i-1|i-1) \end{aligned} \tag{78}$$

and with the notation

$$\boldsymbol{q}_{edj}(i|i-1) = \boldsymbol{F}\boldsymbol{q}_{edj}(i-1|i-1), \tag{79}$$

(78) implies (62). This concludes the proof.

Remark 2. *It is clear that each of Eqs. (59)–(62) is only bound to the jth node and therefore such correction can be done locally for each sensor. Conversely, the system state prediction and the residual vector must be computed globally by Eqs. (39), (53), (57), (58), (63), and (64), respectively.*

Remark 3. *Obviously, under the above conditions, the distributed realization of the Kalman filter correction step is optimal in the sense of criterion (26), and therefore the structure of the fault residual generator based on distributed Kalman filtration is optimal.*

4. Illustrative examples

4.1. Example 1

To eliminate specific system dependencies, the Schur discrete-time linear strictly positive system [23] is used for demonstration of the Kalman filtering technique in residual signals construction. The considered system can be put in the system class (1)–(4), with the sampling period $t_s = 0.8s$, with uncorrelated system and measurement Gaussian noise and the noise covariance matrices

$$\boldsymbol{R} = \text{diag}\;[0.003\;\;0.04], \quad \boldsymbol{Q} = 0.002\boldsymbol{I}_4$$

while the system matrix parameters are

$$\boldsymbol{F} = \begin{bmatrix} 0.7650 & 0.6267 & 0.6058 & 0.0510 \\ 0.1048 & 0.1083 & 0.0813 & 0.0098 \\ 0.1484 & 0.1419 & 0.1171 & 0.0150 \\ 0.1709 & 0.2286 & 0.1603 & 0.1998 \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} 0.0241 & 0.0139 \\ 0.0151 & 0.0013 \\ 0.0109 & 0.0056 \\ 0.0142 & 0.0032 \end{bmatrix}, \boldsymbol{C} = \begin{bmatrix} 0.0001 & 0 & 1 & 0 \\ 0.0000 & 0 & 0 & 1 \end{bmatrix}.$$

Since the discrete-time stochastic linear strictly positive system is stable, the system control law in simulations is defined for the forced mode control as $\boldsymbol{u}(i) = \boldsymbol{W}\boldsymbol{w}_o$, where

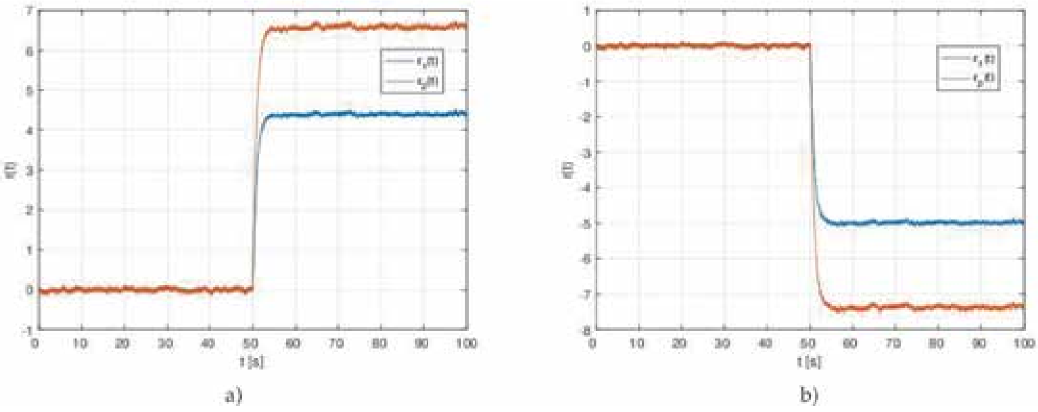
$$\boldsymbol{W} = \begin{bmatrix} -117.3841 & 79.3124 \\ 280.8078 & -187.1829 \end{bmatrix}, \quad \boldsymbol{w}_o = \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}$$

and the initial conditions for the Kalman filter are

$$\boldsymbol{q}_e(0|0) = \boldsymbol{0}, \quad \boldsymbol{P}(0|0) = \boldsymbol{I}_4.$$

Using the given initial conditions, **Figures 1** and **2** display the residuals obtained by the residual filter generated by the distributed Kalman filter defined in (46)–(51), reflecting single actuator and sensor faults, starting at the time instant $t = 50\;\;s$. The time scale is discrete with the sampling period $T = 0.8s$.

Evidently, the residual trajectories indicate that the proposed residual filter generates directional signals in the event of single sensor faults, and has a significantly higher dynamic signal range in the event of single faults of the actuators, as compared to the residual presented using the standard Kalman filter.



a) b)

Figure 1. Residual responses to single faults: (a) the first actuator and (b) the second actuator.

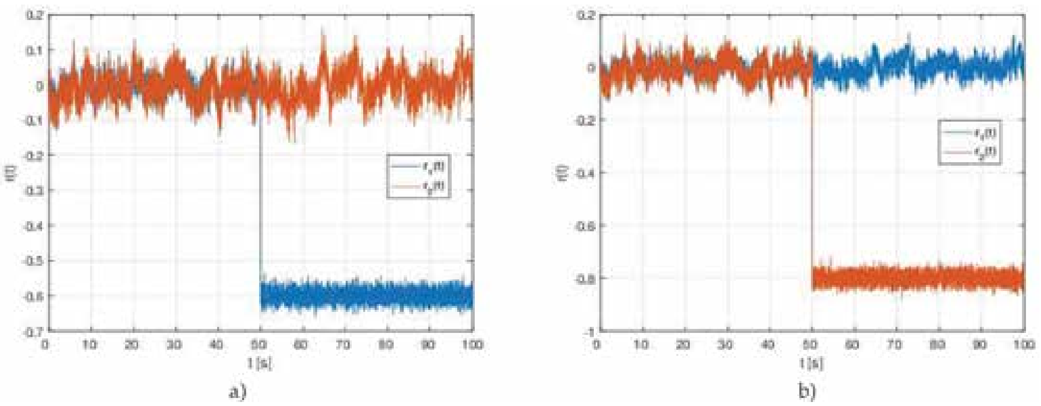


Figure 2. Residual responses to single faults: (a) the first sensor and (b) the second sensor.

4.2. Example 2

To produce another example that demonstrates achievable performances of the presented design method, the sign-indefinite Schur discrete-time linear system is used, where [24]

$$\boldsymbol{F} = \begin{bmatrix} 1.1039 & -0.2360 & -0.0563 & -0.0229 \\ 0.1063 & 0.7971 & -0.0575 & -0.0109 \\ 0.0100 & -0.0211 & 0.9401 & -0.0476 \\ 0.0599 & -0.0843 & -0.0111 & 0.9633 \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} 0.1957 & 0.2878 \\ 0.0976 & 0.1921 \\ 0.0969 & 0.0939 \\ 0.0012 & 0.0982 \end{bmatrix}, \boldsymbol{C} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

$t_s = 0.05s$, and the Gaussian noise covariance matrices are $\boldsymbol{R} = \text{diag}\ [0.003\ \ 0.04]$ and $\boldsymbol{Q} = 0.002\boldsymbol{I}_4$. To force the desired system output values, it is prescribed

$$\boldsymbol{W} = \begin{bmatrix} -2.1250 & 0.9375 \\ 1.8750 & -0.5625 \end{bmatrix}, \quad \boldsymbol{w}_o = \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}, \quad \boldsymbol{q}_e(0|0) = \boldsymbol{0}, \quad \boldsymbol{P}(0|0) = \boldsymbol{I}_4.$$

Figures 3 and **4** present the residual responses of the residual filter based on distributed Kalman filtering, from which it is clear that the used principle, especially when compared to the achievable responses with the alternative system presented in Example 1, is operational. Evidently, the residual filter behavior is also acceptable for the system parameters in this example and the system noise environment. The step-like single faults start and continue from the time instant $t = 25\ \ s$, the time scale is discrete with the sampling period $T = 0.05s$.

4.3. Example 3

Following the above-mentioned procedures, Example 3 verifies their effectiveness for the linear discrete-time system with the parameters

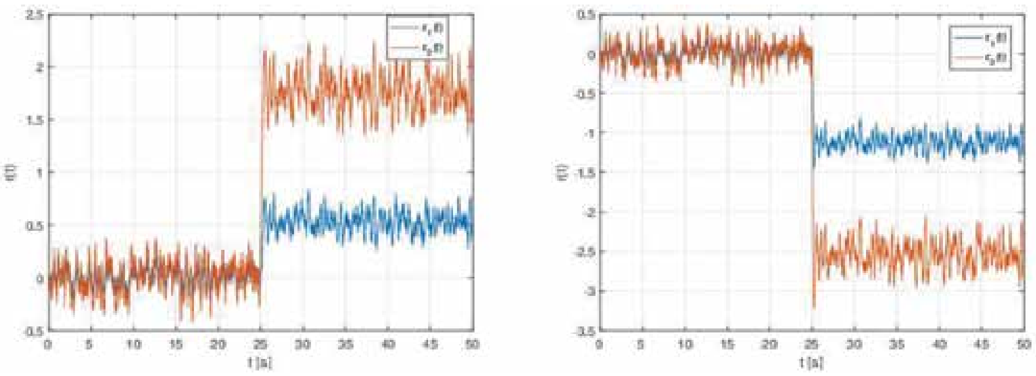


Figure 3. Residual responses to single faults: (a) the first actuator and (b) the second actuator.

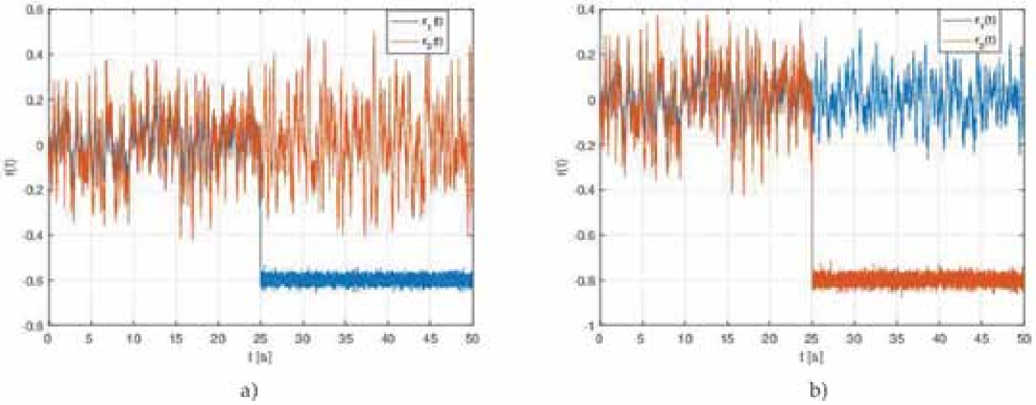


Figure 4. Residual responses to single faults: (a) the first sensor and (b) the second sensor.

$$\mathbf{F} = \begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 \\ 0.7 & 0 & 0 & 0.1 \\ 0.2 & 0.8 & 0.2 & 0.3 \\ 0 & 0 & 0.5 & 0.2 \end{bmatrix}, \boldsymbol{g} = \begin{bmatrix} 0.1 \\ 0 \\ 0.3 \\ 0 \end{bmatrix}, \boldsymbol{g}_f = \begin{bmatrix} 0 \\ 0 \\ 0.3 \\ 0 \end{bmatrix}, \boldsymbol{c}^T = [1 \quad 0 \quad 0 \quad 1], \boldsymbol{c}_f^T = [0 \quad 0 \quad 0 \quad 1],$$

where $\boldsymbol{F}$ is a left-stochastic matrix [25], $t_s = 0.05s$, and the Gaussian noise covariance matrices are $R = 0.003$ and $\boldsymbol{Q} = 0.002\boldsymbol{I}_4$. The behavior of the system is changed by the state-feedback control

$$u(i) = -\boldsymbol{k}^T\boldsymbol{q}(i) + Ww_o$$

where

$$\boldsymbol{k}^T = [0.2982 \quad 1.0731 \quad 0.3711 \quad 0.6412], \quad W = 1.1834, \quad w_o = 0.6,$$

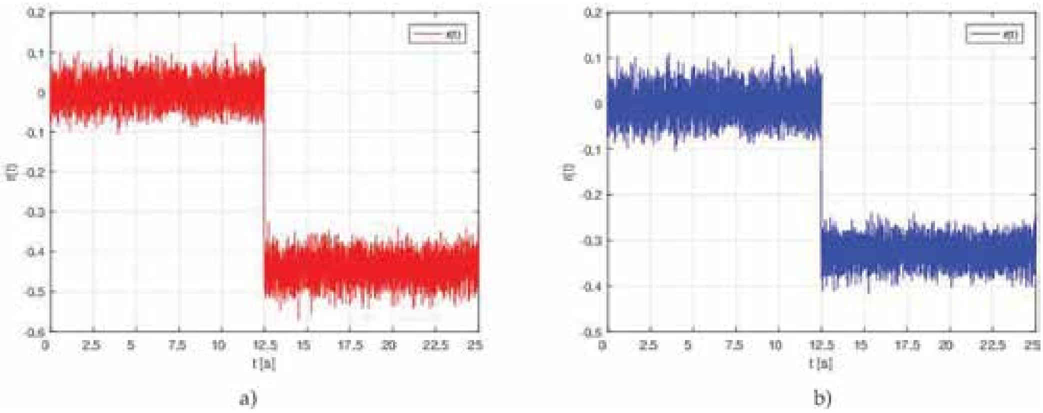


Figure 5. Residual responses to single faults: (a) the actuator and (b) the sensor.

which provides Schur matrix $\boldsymbol{F}_c$ as follows

$$\boldsymbol{F}_c = \boldsymbol{F} - \boldsymbol{g}\boldsymbol{k}^T = \begin{bmatrix} 0.0702 & 0.0927 & 0.2629 & 0.3359 \\ 0.7000 & 0 & 0 & 0.1000 \\ 0.1105 & 0.4781 & 0.0887 & 0.1076 \\ 0 & 0 & 0.5000 & 0.2000 \end{bmatrix}$$

Note, the steady states of $\boldsymbol{F}_c$ are absorbing states.

The single fault effects in residuals, when using the proposed algorithm of distributed Kalman filtering (46)–(51) with setting $\boldsymbol{F} = \boldsymbol{F}_c$, $\boldsymbol{q}_e(0|0) = \boldsymbol{0}$ and $\boldsymbol{P}(0|0) = \boldsymbol{I}_4$ are shown in **Figure 5**. The time scale is discrete with the time sample interval $T = 0.1$s.

From the figures, we find that the fault responses are satisfactory by using the proposed method also for this system and noise environment.

Analyzing all examples, exactly the same responses are reached using the same parameters as before and assuming the same fault patterns if the residuals are evaluated exploiting formulas (57)–(64) or (46)–(51). It is given by the equivalent principles of distributed computing that are bound by the equivalent relationships (44) and (62), respectively. As a result, in this particular point of view, the proposed distributed algorithm has only one common matrix component, $\boldsymbol{P}^{-1}(i|i-1)$, which has to be transmitted to every separated sensor before carrying out the state correction filtering step at every time instant. Since the correction step at time instant i is done in dependency on the measured value at the same time instant $\boldsymbol{y}(i)$, it is clear that the shorter the computation at the correction step, the smaller the time-delay introduced into the fault detection system responses.

5. Concluding remarks

Realization forms for fault detection residual structures, based on distributed Kalman filtering destined for noisy discrete-time linear systems, were derived in this chapter. The main idea deals with introducing a distributed sensor measurement noise corrector step of a Kalman filter, applied in such a way to be locally uncorrelated with other sensor measurements. Two different algorithmic supports, a parallel decentralized Kalman filter and a locally distributed Kalman filter, are constructed to generate fault residuals. Both solutions are discussed in detail to demonstrate the condition of their equivalency. The problem accomplishes the manipulation in the manner giving guaranty of asymptotic stability of a local fault residual detection filter. Simulated example is included to illustrate the applicability of the proposed methods, encouraging the results that are obtained. Note, since the Kalman filter is based on the nominal system parameters $\boldsymbol{G}$ and $\boldsymbol{C}$, it cannot estimate system states and outputs starting for faulty regimes with modified matrices $\boldsymbol{G}_f$ and $\boldsymbol{C}_f$, respectively.

From the point of cloud-based distributed systems, to combine appropriately the network and computational resources, a locally distributed Kalman filter seems to be naturally adaptable, also with cross-correlated sensor noises. Of course, no theoretical justification for this affirmation is presented in the chapter. This is seen as an area for future research by the authors.

Acknowledgements

The work presented in this chapter was supported by VEGA, the Grant Agency of the Ministry of Education and the Academy of Science of Slovak Republic, under Grant No. 1/0608/17. This support is very gratefully acknowledged.

Author details

Dušan Krokavec* and Anna Filasová

*Address all correspondence to: dusan.krokavec@tuke.sk

Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Košice, Slovakia

References

[1] Hassan MF, Salut G, Singh MG, Titli A. A decentralized computational algorithm for the global Kalman filter. IEEE Transactions on Automatic Control. 1978;**23**(2):262-268

[2] Song E, Zhu Y, Zhou J, You Z. Optimal Kalman filtering fusion with cross-correlated sensor noises. Automatica. 2007;**43**(8):1450-1456

[3] Hajiyev C, Caliskan F. Sensor/actuator fault diagnosis based on statistical analysis of innovation sequence and robust Kalman filtering. Aerospace Science and Technology. 2000;**4**(6):415-422

[4] Hu J, Xiao L. Multi-sensor fault diagnosis of aircraft engine based on Kalman filter group. In: Proceedings of 2016 Chinese Intelligent Systems Conference, Vol. 1; 22–23 October 2016; Xiamen, China; Lecture Notes in Electrical Engineering 404. Singapore: Springer Science + Business Media; 2016. pp. 363-379

[5] Mehra RK, Peschon J. An innovations approach to fault detection and diagnosis in dynamic systems. Automatica. 1971;**7**(512):637-640

[6] Petersen IE, Savkin AV. Robust Kalman Filtering for Signals and Systems with Large Uncertainties. Boston: Birkhäuser; 1999. 207 p

[7] Kobayashi T, Simon DL. Evaluation of an enhanced bank of Kalman filters for in-flight aircraft engine sensor fault diagnostics. Transactions of the ASME. 2005;**127**(3):497-502

[8] Xue W, Guo YQ. Application of Kalman filters for the fault diagnoses of aircraft engine. In: Kordic V, editor. Kalman Filter. In Tech: Rijeka; 2010. pp. 349-362

[9] Liu X, Xue N, Yuan Y. Aircraft engine sensor fault diagnostics using an on-line OBEM update method. PLoS One. 2017;**12**(2):1-23

[10] Zhu Y, Zhou J, Shen X, Song E, Luo Y. Networked Multisensor Decision and Estimation Fusion Based on Advanced Mathematical Methods. Boca Raton: CRC Press; 2013. 437 p

[11] Duda Z. State estimation in a decentralized discrete time LQG control for a multisensor system. Archives of Control Sciences. 2017;**27**(1):29-39

[12] Filasová A, Krokavec D. On distributed discrete-time Kalman filtering in large linear time-invariant systems. In: Proceedings of the 21st International Conference on Process Control PC; 6–9 June 2017; Štrbské Pleso, Slovakia. 2017. pp. 291-296

[13] Mahmoud MS. Decentralized Control and Filtering in Interconnected Dynamical Systems. Boca Raton: CRC Press; 2017. 612 p

[14] Govaers F. Distributed Kalman filter. In: Serra GL, editor. Kalman Filters. Theory for Advanced Applications. London: IntechOpen; 2018. 314 p

[15] Speyer JL. Computation and transmission requirements for a decentralized Linear-Quadratic-Gaussian control problem. IEEE Transactions on Automatic Control. 1979; **24**(2):266-269

[16] Berg TM, Durrant-Whyte HF. General decentralized Kalman filters. In: Proceedings of the American Control Conference, Vol. 2; 29 June–1 July 1994; Baltimore, USA. 1994. pp. 2273-2274

[17] Zhu Y. Multisensor Decision and Estimation Fusion. New York: Kluwer Academic Publishers; 2003. 236 p

[18] Marelli D, Zamani M, Fu M, Ninness B. Distributed Kalman filter in a network of linear systems. Systems & Control Letters. 2018;**119**:71-77

[19] Chui CK, Chen G. Kalman Filtering with Real-Time Applications. Berlin: Springer-Verlag; 2009. 230 p

[20] Krokavec D. Decentralized optimal filtering of the large-scale system state. In: Proceedings of the 2nd IFAC Workshop New Trends in Design of Control Systems; Smolenice, Slovakia, 7–10 September 1997; Volumes 30(21). 1997. pp. 445-450

[21] Horn RA, Johnson CR. Matrix Analysis. New York: Cambridge University Press; 2013. 644 p

[22] Bryson AE, Ho YC. Applied Optimal Control. Optimization, Estimation, and Control. New York: Taylor & Francis; 1975. 482 p

[23] Krokavec D, Filasová A. Stabilization of discrete-time LTI positive systems. Archives of Control Sciences. 2017;**27**(4):575-594

[24] Xue D, Chen Y, Atherton DP. Linear Feedback Control. Analysis and Design with MATLAB. Philadelphia: Society for Industrial and Applied Mathematics; 2007. 370 p

[25] Canto B, Canto R, Kostova S. Stabilization of positive linear discrete-time systems by using a Brauer's theorem. The Scientific World Journal. 2014;**2014**:1-6

Edited by Constantin Volosencu

This book offers a selection of papers in the field of fault detection and diagnosis, promoting new research results in the field, which come to join other publications in the literature. Authors from countries of four continents: United States of America, South Africa, China, India, Algeria and Croatia published worked examples and case studies resulting from their research in the field. Fault detection and diagnosis has a great importance in all industrial processes, to assure the monitoring, maintenance and repair of the complex processes, including all hardware, firmware and software.

The book has four sections, determined by the application domain and the methods used: 1. Hybrid Computing Systems, 2. Power Systems, 3. Power Electronics and 4. Kalman Filtering. In the first section, the readers will find a technical report on fault diagnosis of hybrid computing systems, based on the chaotic-map method that uses the exponential divergence and wide Fourier properties of the trajectories, combined with memory allocations and assignments. In the second section, two chapters are included: one of them presents a study on preventive maintenance and fault detection for wind turbine generators using statistical models and the second chapter presents a technical report on fault diagnosis for turbo-generators, based on the mechanical-electrical intersectional characteristics. The third section contains a technical report that presents some techniques of detection and localization of open-circuit faults in a three-phase voltage source inverter fed induction motor. The fourth section presents a theoretical study on the application of distributed discrete-time linear Kalman filtering with decentralized structure of sensors in fault residual generation.

Published in London, UK

© 2018 IntechOpen
© Dmitrii Kotin / iStock

IntechOpen

ISBN 978-1-83881-831-9