Embedded anomaly detection - Machine learning based radiation hardening of space electronics

Adrien Dorise

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)*

Présentée et soutenue le *02/12/2022* **par :**

Adrien DORISE

Embedded anomaly detection
Machine learning based radiation hardening of space electronics

JURY

Vincent COCQUEMPOT	Professeur d'Université	Président du Jury
Cédric RICHARD	Professeur d'Université	Rapporteur
Frédéric WROBEL	Professeur d'Université	Rapporteur
Núria AGELL	Professeur d'Université	Membre du Jury
François VACHER	Ingénieur de recherche	Membre du Jury
Audine SUBIAS	Professeur d'Université	Co-encadrante de thèse
Louise TRAVÉ-MASSUYÈS	Directeur de recherche	Directrice de thèse
Corinne ALONSO	Professeur d'Université	Co-directrice de thèse

École doctorale et spécialité :
EDSYS : Informatique 4200018
Double mention :
EDSYS : Systèmes embarqués 4200046
Unité de Recherche :
Laboratoire d'analyse et d'architecture des systèmes LAAS-CNRS (UPR8001)
Directeur(s) de Thèse :
Louise TRAVÉ-MASSUYÈS et *Corinne ALONSO*
Rapporteurs :
Cédric RICHARD et *Frédéric WROBEL*

Acknowledgements

Si ces trois années de vie scientifique m'ont donné la possibilité de mieux comprendre les enjeux de la recherche, elles m'ont aussi permis de rencontrer de nombreuses personnes qui m'ont guidé, inspiré et épaulé, et que j'aimerais remercier.

Premièrement, je voudrais remercier mes directrices de thèse Louise TRAVÉ-MASSUYÈS, Corinne ALONSO et Audine SUBIAS pour leur soutien et leur accompagnement durant cette thèse. Elles ont su m'aiguiller sur ce sujet pluridisciplinaire tout en me laissant libre de choisir la direction de mes travaux. Je remercie également mes encadrants CNES François VACHER, Leny BACZKOWSKI et Thomas TORLOTING pour leurs conseils et leur soutien au cours de cette thèse. Je remercie les équipes DISCO, et ISGE pour m'avoir accueilli.

Je remercie Cédric RICHARD et Frédéric WROBEL pour avoir accepté de rapporter mon manuscrit de thèse. Les remarques et commentaires étaient très intéressants, et ont permis de situer mes travaux tant bien dans le domaine des radiations spatiales, que dans celui de la détection d'anomalies. Je remercie également les membres du jury Vincent COCQUEMPOT et Núria AGELL pour la pertinence de leurs questions, et la bienveillance de leurs commentaires.

Je tiens à remercier François BEZERRA et Robert ECOFFET pour m'avoir guidé et m'avoir transmis leurs connaissances sur le domaine des radiations. Un grand merci à Françoise pour son aide dans la récolte des données, que ce soit en me donnant accès à des bancs de test, ou en me permettant d'utiliser les données d'anciens tests.

Je tiens à remercier chaleureusement toutes les personnes qui sont devenues plus que des collègues au cours de ces trois années. Je remercie Mathieu, qui m'a permis de voir de mes propres yeux que la perfection existe en ce monde. J'ai été très heureux d'assister à ton mariage, et je te souhaite le meilleur avec Elsa. Je remercie Camille, avec qui j'ai vécu en quasi-colocation pendant ces périodes de pandémies. J'ai adoré discuter de tout et de rien avec toi. Tu as toutes les qualités pour devenir un grand chercheur, et je te souhaite le meilleur pour la suite. Je remercie Yoni, mon plus grand adversaire de ping-pong, devenu un merveilleux ami. J'ai hâte d'assister à ta thèse. Je remercie Eric, un grand homme, avec qui les souvenirs de soirées sont étrangement flous. Je remercie Corbi, Claire, Mauro, Nicolas, Alexandre, Le Toan, Charlotte, Edgar, Amaury, Fréd C., Edouard, Fréd G., Soheib, Pauline... pour tous ces moments passés au sein du laboratoire.

Une thèse se passe également en dehors du laboratoire. J'aimerais donc remercier chaleureusement mes comparses musicaux, Carole et Maxence pour tous ces moments passés ensemble.

J'aimerais remercier les personnes qui m'ont marqué au cours de ma vie, et qui quelque part, ont permis l'aboutissement de ces travaux. Je remercie Ronan, que j'ai eu la chance de rencontrer à plusieurs milliers de kilomètres de mon pays, et dont le destin nous a réunis sur Toulouse pour un doctorat, chacun dans son domaine. Je remercie chaleureusement Vincent, Stan, Antoine et Julie.

J'ai également la chance d'avoir été accompagné par une famille qui m'a toujours

soutenu, et qui même sans comprendre, a toujours montré de l'intérêt pour ce que je faisais (non Papi, les nuages n'empêchent pas de communiquer avec les satellites). Je remercie énormément mon Père, ma Mère, et ma Sœur, avec qui j'aurai l'immense joie de partager le titre de Docteur, même si moi, je ne sauve pas des vies.

Enfin, j'adresse mes derniers remerciements à celle sans qui cette thèse n'aurait jamais abouti. Celle qui a toujours été là dans les moments les plus difficiles, celle qui a toujours été là pour écouter mes échecs et mes découvertes, celle qui a changé ma vie depuis qu'elle y est rentrée. Merci Julia.

Pour conclure, si la vie est un long fleuve tranquille, le doctorat se compare aux chutes du Niagara.

Abstract

The miniaturisation of electronic components is one of the major improvements that happened during the last decades. Space agencies followed this trend, and satellites became more and more compact, while their embedded functions increased their complexities. Unfortunately, by reducing the size of their components, satellites became subject to space radiation. Indeed, satellites are not protected by the terrestrial atmosphere and defects caused by energetic particles can happen. These events are called "Single Event Effect" (SEE). Their consequences range from a functional interruption of the component to its destruction. Therefore it is essential to implement solutions to protect the satellites against SEEs. However, SEEs are random and can take many forms, so detecting and preventing them is a scientific challenge. Nowadays, protection methods are based on hardening methods, that modify the fabrication process of a component. The major inconvenience is that hardened components are complex to design, and heavy engineering studies are mandatory.
As a consequence, the price of hardened components increases significantly. However, since the 21^{st} century and the rise of the new space era, the use of hardened components has decreased significantly. Alternatively, a threshold protection method exists to prevent destructive single event effects from harming electronic components. However, not all faults can be detected by the sole use of this method. Researches on new techniques need to be developed to detect single event effects.

Anomaly detection is a sub-field of artificial intelligence and machine learning. Its aims to detect patterns that deviate from a well-defined normal behaviour. To do so, a model is trained on known data to be able to generalise and predict the system's future behaviour. It is then possible to pinpoint abnormal observations occurring in the system. These approaches are called data-based methods, and differ from model-based approaches, as they do not require an overall knowledge of the system. These methods proved to be efficient in many fields, but are yet to be applied to single event effects detection.

This manuscript details research work done to evaluate the performance of machine learning on the detection of single event effects for space applications. For this purpose, two distinctive types of research were conducted. The first research is focused on single event effects characteristics and their potential impacts on an ATMEL SAM3X8E microcontroller. A significant database has been created, mixing both experimental and simulated data. In addition, a thorough analysis of the impact of single event effects on the supply current is made to extract the most meaningful features.
Following the groundwork laid by the first research, machine learning performance is thoroughly studied for the detection of single event effects in a second research work. A proof of concept is realised by using most common anomaly detection methods. It is demonstrated that the performance of machine learning is on par with today's threshold methods, outperforming it for the detection of non-destructive single event effects. From there, a specific algorithm is developed. Called Dynamic Double anomaly Detection (DyD^2), this algorithm meets space applications requirements. Extensive experiments have been conducted to test DyD^2 on simulated data, as well as real-time on-board application with the ATMEL SAM3X8E, that proved that DyD^2 is a valid alternative to today's detection methods.

Artificial intelligence, Machine learning, Anomaly detection, Embedded systems, Space applications, Radiation

Résumé

Les satellites n'étant pas protégés par l'atmosphère terrestre, ils sont soumis aux radiations spatiales. L'un des effets de ces radiations est l'apparition de perturbations liées à des particules isolées, allant de l'arrêt temporaire du composant, jusqu'à sa destruction. Il est donc important de protéger les composants électroniques utilisés lors d'une mission spatiale. Jusqu'à présent, la méthode la plus commune se base sur le durcissement des composants : un composant durci voit sa conception modifiée pour se prémunir des perturbations liées à une particule isolée, appelées événements singuliers. Cette solution nécessite une ingénierie complexe, qui se répercute sur le prix des composants. Avec la nouvelle ère spatiale qui a débuté au $21^{ème}$ siècle, l'utilisation de composants durcis diminue, au profit de composants moins coûteux. Une alternative au durcissement consiste en une méthode de détection de seuil de sur-intensité et/ou de sur-tension. Plus générale et moins coûteuse, cette méthode ne permet cependant pas de détecter tous les types de défauts. De ce fait, de nouvelles méthodes sont nécessaires pour préserver l'intégrité des composants lors de missions spatiales.

La détection d'anomalies est un domaine appartenant à l'intelligence artificielle et à l'apprentissage automatique. L'objectif est de mettre en évidence des comportements qui dévient du fonctionnement nominal d'un système. Pour ce faire, un modèle est entraîné sur des données d'apprentissage afin de modéliser et de prédire le comportement du système. À partir de ce modèle, il est possible d'identifier les comportements anormaux, et d'agir en conséquence. Cette approche, basée sur les données, se distingue par le fait qu'elle ne nécessite pas de connaissances préalables du système étudié. Aussi, avec la montée en popularité de ces méthodes, de nombreuses applications ont démontré l'efficacité de ces méthodes dans le cadre de la détection d'anomalies. Néanmoins, ces méthodes n'ont pas encore été testées pour la détection des événements singuliers

Ce manuscrit détaille le travail réalisé visant à démontrer l'intérêt des méthodes d'apprentissage automatique pour la détection des défauts provoqués par des particules isolées. Dans ce but, notre recherche s'articule en deux parties. La première partie est consacrée à l'impact des radiations sur les composants électroniques et l'étude d'un composant de type microcontrôleur. La seconde partie s'intéresse à l'application des méthodes d'apprentissage automatique pour la détection de défauts dus aux radiations spatiales sur composants électroniques.

La première partie se décompose en deux sections distinctes. Premièrement, une plateforme expérimentale a été créée. Pour ce faire, un composant électronique est sélectionné sur lequel la majorité des études sera menée. Le microcontrôleur ATMEL SAM3X8E est choisi. Sa sensibilité aux radiations, son accessibilité ainsi que le fait qu'il soit actuellement utilisé dans le contexte de missions spatiales sont les trois raisons qui ont conduit à ce choix. Des études de comportements ont été réalisées afin d'établir des profils de normalités. De ces études, un profil de consommation lié au logiciel ainsi qu'un profil de consommation lié aux charges placées sur les I/O du composant ont été mis en évidence. Suite à cela, une carte de test appelée DIAG-RAD a été développée. Cette carte permet la réalisation de tests visant à émuler les effets des radiations sur le microcontrôleur SAM3X8E tout en simulant des scénarios de fonctionnement. Elle répond à diverses spécifications, comme la possibilité de changer facilement de composant de

test en cas de défaillance, ou encore la possibilité de simuler les profils de consommations décris précédemment.

Dans un second temps, une base de données contenant des défauts liés aux radiations est créée. Elle est constituée à la fois d'observations provenant du fonctionnement nominal du SAM3X8E, mais également d'observations d'événements singuliers. Afin de recueillir ces observations de défaillance, trois campagnes de tests sont menées. Des essais de test lasers ainsi que des essais utilisant une source radioactive sont réalisés afin d'émuler des défauts dans le courant de consommation du SAM3X8E. En complément, des essais ions lourds réalisés par le CNES sur le BS62LV4006 CMOS ont été récupérés permettant d'étendre l'étude à d'autres composants. Suite à ces essais, un simulateur de courant est développé sous MATLAB afin de disposer d'une base de données conséquente. Ce simulateur repose sur les données récoltées du courant de consommation du SAM3X8E en fonctionnement normal ainsi que des données récoltées lors des tests expérimentaux. Il est également possible de paramétrer le simulateur afin d'ajouter diverses variations dans le courant simulé, tel que des sauts de courant ou des déviations linéaires afin de créer divers scénarios de comportements.
De plus, une étude sur l'impact des événements singuliers sur le courant de consommation du SAM3X8E est réalisée. Cette étude a été rendue possible grâce aux données collectées lors des différentes campagnes de tests. Lors de cette étude, des indicateurs statistiques ainsi que la signature fréquentielle du courant de consommation sont examinés afin de caractériser les événements singuliers. L'objectif final est d'extraire les attributs les plus pertinents afin d'entraîner les modèles d'apprentissage automatique. Les résultats font ressortir quatre indicateurs statistiques : la moyenne arithmétique, la variance, l'erreur-type de la moyenne et l'écart absolu médian. Ces attributs sont utilisés lors de la seconde partie concernant l'utilisation des méthodes d'apprentissage automatique. De plus, l'analyse du spectre fréquentiel montre qu'il est possible de caractériser les défauts liés aux radiations en analysant les fluctuations des harmoniques. En effet, il a été mis en évidence que l'amplitude des pics de fréquence prédominants en comportement normal diminue, et l'ajout de pics de fréquence bruités sont caractéristiques de l'impact des événements singuliers sur le SAM3X8E.

La deuxième partie de ce manuscrit porte sur l'application des méthodes d'apprentissage automatique pour la détection de défauts sur composant électronique. Trois sections distinctes divisent cette partie. Dans un premier temps, une preuve de concept est réalisée concernant la validité des approches d'apprentissage automatique pour la détection de défauts dus aux radiations spatiales sur des composants électroniques. Pour ce faire, diverses méthodes d'apprentissage automatique parmi les plus connues sont utilisées et testées sur la base de données détaillée précédemment. Dans ces travaux, ces méthodes sont divisées en trois catégories appelées classification, classification renforcée par clustering et classification mono classe. Afin d'analyser la performance de chaque méthode, trois critères de validation sont définis. Ces critères différencient la détection de défauts destructifs, qu'un modèle doit détecter sans faille, de la détection de défauts non-destructifs, qui permettent une marge d'erreur dans la qualité de la détection. Les résultats de ces expérimentations montrent que les méthodes d'apprentissage sont efficaces

dans la détection de défauts dus aux radiations spatiales sur des composants électroniques.

Dans un second temps, suite aux résultats positifs obtenus précédemment, un nouvel algorithme d'apprentissage automatique a été développé afin de répondre aux spécifications du contexte spatial. Ces spécifications incluent une implémentation embarquée de l'algorithme. Cela se traduit par un traitement en temps réel des observations ainsi qu'une exécution de la méthode sur des composants possédant une puissance de calcul limitée. Aussi, du fait du vieillissement des composants accéléré par l'environnement radiatif, la méthode doit pouvoir s'adapter à un environnement dynamique. Enfin, cette méthode ne doit nécessiter que des observations provenant du comportement normal lors de l'entraînement, car il est complexe de collecter des données de défauts.
La méthode ainsi développée est appelée Dynamic Double anomaly Detection (DyD^2). Cette méthode se base sur un clustering dynamique afin de proposer une détection en quatre phases. La première phase permet de repérer les ruptures dans un signal. La deuxième phase met en place une détection rapide des défauts destructifs. La troisième phase, plus lente, permet la détection de défauts non-destructifs qui pourraient être confondus avec le comportement normal. Enfin, la dernière phase met à jour le modèle avec les dernières observations afin de suivre une éventuelle déviation du comportement normal du système.

Dans un troisième temps, DyD^2 est testé sur divers jeux de données afin de vérifier ses performances. DyD^2 est tout d'abord testé sur des jeux de données provenant du simulateur de courant décrit précédemment. Ces résultats ont permis de comparer DyD^2 avec d'autres algorithmes d'apprentissage automatique. Il en ressort que DyD^2 propose des performances similaires aux méthodes provenant de l'état de l'art. De plus, grâce à des tests sur des jeux de données contenant une déviation linéaire, il a été montré que DyD^2 est capable de s'adapter à un système dynamique. Par la suite, DyD^2 a été testé sur des jeux de données provenant de campagnes de tests sous un accélérateur de particules. Il a été montré que les résultats de DyD^2 corroborent ceux obtenus par simulation. Enfin DyD^2 a été embarqué sur microcontrôleur SAM3X8E afin de détecter des défauts provenant d'un deuxième SAM3X8E sous un faisceau laser. Cette expérience a montré que DyD^2 est capable de fonctionner de façon nominale sur des composants possédant de faibles puissances de calcul. Ces résultats ont permis de mettre en évidence que DyD^2 est une alternative efficace aux méthodes de détection actuellement utilisées par l'industrie spatiale.

Intelligence artificielle, Apprentissage automatique, Détection d'anomalies, Systèmes embarqués, Application spatiales, Radiation

Publications

Conferences

- Adrien Dorise, Audine Subias, Louise Travé-Massuyès, Corinne Alonso. Advanced machine learning for the detection of single event effects. Radiation and its Effects on Components and Systems - RADECS 2022, Oct 2022, Venice, Italy. (Accepted)
- Adrien Dorise, Louise Travé-Massuyès, Audine Subias, Corinne Alonso. Dyd^2: Dynamic Double anomaly Detection: Application to on-board space radiation faults. IFAC Safeprocess 2022 :11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes, IFAC, Jun 2022, Pafos, Cyprus. (Published)
- Adrien Dorise, Corinne Alonso, Audine Subias, Louise Travé-Massuyès, Leny Baczkowski, et al.. Machine learning as an alternative to thresholding for space radiation high current event detection. Radiation and its Effects on Components and Systems - RADECS 2021, Sep 2021, Vienna, Austria. (Published)

Workshops

- Adrien Dorise, Louise Travé-Massuyès, Corinne Alonso, Audine Subias, François Vacher, et al.. Creation of a database for single events latch-up detection on Atmel SAM3X microcontroller. European Space Components Conference - ESCCON 2021, ESA, Mar 2021, Online, Netherlands.
- Adrien Dorise, Louise Travé-Massuyès, Corinne Alonso, Audine Subias, François Vacher, et al.. Anomaly Detection for Radiation Hardening of Space Electronics - Application of Machine Learning Algorithms on an Atmel SAM3X Microcontroller. Avionics, Data, Control and Software Systems - ADCSS 2020, ESA, Oct 2020, Amsterdam (virtual), Netherlands.

Oral communication

- Adrien Dorise, Louise Travé-Massuyès, Audine Subias, Corinne Alonso. Dynamic Double anomaly Detection through evolving clustering: Application to on-board space radiation faults. ANITI presentation, Mar 2022, Toulouse, France. 2022.

Contents

List of Figures

List of Tables

Part I

Introduction

Chapter 1
Introduction

1.1 Context/Motivation

Since the launch of the first satellite Sputnik-1 on the 4th of October 1957, the space industry has become part of society. Communication, topographic pictures, weather forecast and scientific research are heavily dependent on the few three thousand satellites in operation around Earth. With the growth in demand and the advent of the new space era, the number of launches grew drastically.
New technological constraints emerged from the outer space hostile environment. Indeed, space electronic components are subject to collisions with high energetic particles. The faults resulting from these encounters are called "single event effects". It can be tricky to counter such faults as these phenomena are random and take various forms. Designing protection against single event effects is called "hardening". The most common method consists in modifying a component during its conception. Radiation-hardened components are based on their non-hardened equivalents, with some design and manufacturing variations that reduce the susceptibility to radiation damage. Research of new hardening techniques is a major topic for the space industry as it directly impacts mission reliability and costs.

With the emergence of machine learning, the industry gradually shifted and new solutions appeared. In the space industry, machine learning is core in many applications and showed highly beneficial for image processing. However, machine learning, and mostly anomaly detection methods, are yet to be tested for the protection of space components.

1.2 Goal

The primary goal of this thesis is to propose new solutions for the detection of single event effects. Indeed, this thesis project aims at exploring machine learning algorithms that could replace traditional hardening methods. The work has been organized along the three following sub-goals:

- Building a database gathering both normal and radiation faults observations.
- Providing a proof of concept to establish the feasibility of machine learning methods.
- Proposing a novel machine learning based method for anomaly detection accounting for on-board constraints.

1.3 Thesis content

One of the particularities of this work is its multidisciplinary nature. Tackling radioactivity, electrical engineering and computer science, a large variety of knowledge is needed to get a good insight of this project. Therefore, it has been decided to focus on two aspects: radiation effects and anomaly detection.

Part I describes the theoretical background of this work.
In chapter 2 the specificities of the space environment are reported. A thorough review of the different kinds of radiation is given, as well as their effect on electronic devices.
In chapter 3 machine learning and its application for anomaly detection are detailed. An insight of the different types of anomalies and algorithms used in this field is given.

Part II is dedicated to the creation of databases
In chapter 4, the components and the hardware used during this thesis are described. An overview of the SAM3X8E microcontroller is given as well as a description of the DIAG-RAD electronic board specifically designed for this thesis project are given.
In chapter 5, the focus is on the creation of an extensive database for the study of single event effects. A description of the various experiments done during this project is given. Also, a supply current simulator developed for the needs of this thesis is detailed. Finally, a thorough study of the features characterising a single event effect is performed.

Part III is dedicated to the application of machine learning to radiation fault detection.
In chapter 6, a proof of concept is performed to evaluate the validity of machine learning approaches. State of art algorithms are tested on various databases containing single event effects.
In chapter 7, a new algorithm specifically designed for the detection of single event effects and meeting the space industry requirements is proposed. Called Dynamic double Anomaly Detection (DyD^2), it is the main contribution of this thesis.
In chapter 8, an evaluation of the performance of DyD^2 using different data sets is performed. These tests are made to evaluate the performance of DyD^2 in the case of real space applications.

Chapter 2
Space environment

Contents

Unlike most applications that are protected by the Earth's atmosphere, the space industry has to deal with specific constraints linked to radiation. Space components are subject to high levels of particle bombardment throughout their lifetime, which can endanger the success of a mission. Therefore, a solid understanding of the different phenomena at hand enables researchers to develop countermeasures.

This chapter focuses on introducing the space environment, and more particularly the radiation activity involved around embedded applications. It resolves around two questions:

- *How does radiation impact electronic components and embedded applications?*
- *How can protection be designed?*

First, a little bit of context regarding the space industry is given in section 2.1. Then section 2.2 describes the different sources of radiation found in space that are likely to damage space components. After that, section 2.3 focuses on the various impacts caused by radiation on electronic components. Following in section 2.4, the different methods used to emulate radiation faults on Earth are described. Finally, section 2.5 briefly describes some techniques used to protect electronic devices against the effects presented beforehand, in order to position our work regarding the already existing solutions.

2.1 Space industry context

Moore's law states that the number of transistors in integrated circuits (IC) doubles yearly, improving computers' performance in the process [1]. Despite being an unproven theory, this statement has proved to be true and led the semiconductor industry for many decades. Indeed, the growth in computation power has evolved significantly since the introduction of Moore's law in 1965. With the massive increase in computation power, applications became more complex and led to many technological breakthroughs.

However, the harsh space environment and the limited satellite size are factors that render the use of *COTS* (Commercial Off-The-Shelf) components for the space industry complicated. Moreover, the impossibility of undergoing maintenance means that the failure of a component often induces the end of the mission. Therefore, extensive research is conducted on components to improve their reliability in a space environment. This process is called *hardening*.

With the advent of SpaceX and the successful launch of its reusable launcher Falcon9 in 2010, a new revolution emerged in the space industry. It was characterised by a massive increase in the use of COTS, leading to a drastic drop of the cost of space missions, although lowering the reliability of the components. This space revolution is known as the *new space* era. In the meantime, more and more satellites were launched each year, as well as the number of mission failures. Precisely, from temporary loss of communication with a satellite, to complete destruction of one of its components, these disruptions have become more frequent since 2010. From around 15% between 2000 and 2008, the percentage of failure skyrocketed to around 35% between 2012 and 2016 [2]. It is why this thesis work aims at improving the reliability of space components at reduced cost. By doing so, it would be possible to decrease missions failures while providing low cost solutions to the space agencies.

One of the leading causes of failures comes from space radiation. For this reason, it is essential to get an insight into the different types of radiation, as well as their impact on electronic devices.

2.2 Cosmic radiation

Before diving into the radiation sources, it is important to get a fundamental insight into radioactivity. Radioactivity is energy released by the disintegration of naturally unstable atoms seeing their electrons stripped away. The energy released from this separation can be emitted in the form of rays, electromagnetic waves and particles. On Earth, sources of radiation are numerous. From natural radiation coming from various materials, to man-made radiation such as nuclear plants, X-rays, microwaves or cellphones.
However, most radiation sources are not coming from Earth, but instead are coming from space. These radiations are actively blocked by the magnetic field surrounding Earth, and few reach the ground. The International Atomic Energy Agency (IAEA) defines *cosmic radiation* as radiation received from outer space. These radiations can be divided into solar flares, radiation belts, and galactic cosmic rays. Each of them is going to be thoroughly described in the following sections, but a quick summary is available table 2.1.

Source	Particle type	Energy (in electronvolt (eV))
Radiation belt	Protons Ions	few keV to 500 MeV few eV to 10 MeV
Galactic cosmic rays	Ions Atomic nuclei	From 1GeV to 10^8 TeV
Solar flares	Protons Electrons Heavy ions	few keV to 500 MeV few keV to 500 MeV 1-10 MeV/n

Table 2.1: Radiation types (data from [3])

2.2.1 Radiation belts

Radiation belts are the most problematic type of radiation for space applications, as it is the closest source of radiation to satellite orbits. Radiation belts result from trapped energetic particles by a planet's magnetic field. Earth possesses two radiation belts called *Van Allen radiation belts*, named after James Van Allen, who discovered them in 1958 [4]. It is in January 1958, that Explorer 1 was launched by Van Allen from Cap Canaveral, equipped with Geiger-Müller tubes to measure radiative activity. However, at the time, the radiation levels in space were heavily undervalued, leading to a saturation of most of the instruments. Nevertheless, Van Allen still achieved to conclude that the radiation field inside the satellite's components was around 0.06 rad per hour. As the maximum dose rate fixed for a human being is approximately 0.3 rad per **week**, we can easily imagine what would have happened to Yuri Gagarin, the first human to reach outer space in 1961, without those discoveries.

2.2.2 Solar flares

Also called *Solar Particle Event (SPE)*, solar flares are massive amounts of energy released by the sun. It is the most potent magnetic event in the solar system [5]. Solar storms consist of a series of solar flares that can heavily damage electronic components by the high quantity of energetic particles released at once, disturbing space missions. For example, the Halloween solar storm in October 2003 significantly impacted satellite systems and communication. In addition, this excess of particles hitting Earth's magnetic field resulted in aurorae visible in uncommon latitudes, such as Texas.

Even though solar flares were known in China centuries before Christ, it was only with the invention of the telescope in the 18th century that solar activity started to be extensively studied [6]. These observations led to the discovery of an 11-year solar cycle that helped astronomers predict solar activity [7]. In addition, these predictions helped space agencies to adapt their mission to solar activities. The Sunspot Index and Long-Term Solar Observations (SILSO) make available a public record of the sunspot activity. In figure 2.1, the 11 years cycle as well as a slower 100 years cycle can be identified from Sunspot data measured since 1750.

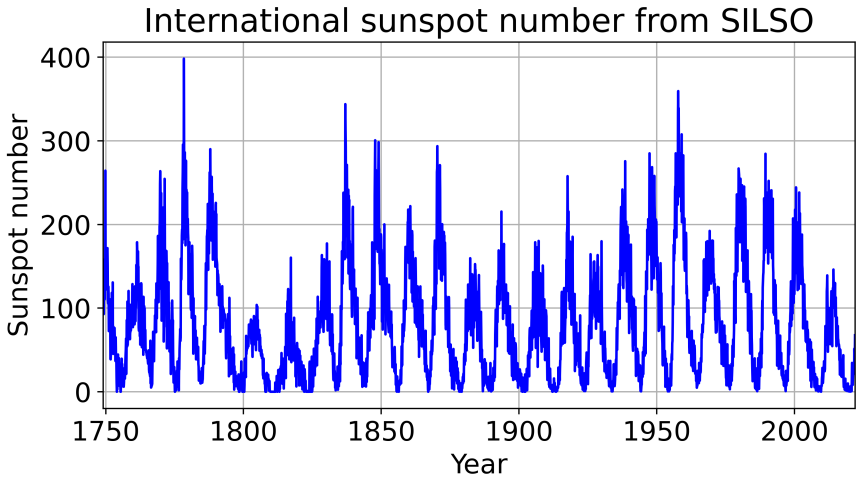


Figure 2.1: Sunspot observation since 1750 (data from [8])

2.2.3 Galactic Cosmic Radiation (GCR)

Galactic cosmic radiations are coming from outside our solar system. They are constituted by nuclei of atoms travelling in space at near light speed. Events such as the end of life of a massive star, resulting in a powerful explosion called supernova, can accelerate those nuclei. They are incredibly high-energy particles that can easily ram and even cross space equipment causing severe damage to space equipment, going as far as total destruction of the equipment. Note that in our solar system, they can be repelled by the sun's magnetic field. Therefore, their intensity is increased during low solar activities.

GCR are in majority absorbed by Earth's atmosphere that is acting as a powerful protection. When cosmic radiation impacts Earth's atmosphere, a shower of secondary particles is produced, creating additional particles of various natures, such as proton, neutron or photon, as displayed in figure 2.2.

This phenomenon was discovered by Victor Hess in 1912 with balloon experiments [9]. At the time, only radiations coming from Earth's soil were known, and expectations were that radiation levels would decrease at high altitudes. However, Hess's experiments gave different results, as even though radiation level first decreased with altitude, it was not without a certain surprise that the level rose again at a higher altitude. Hess predicted correctly that this was due to cosmic radiation.

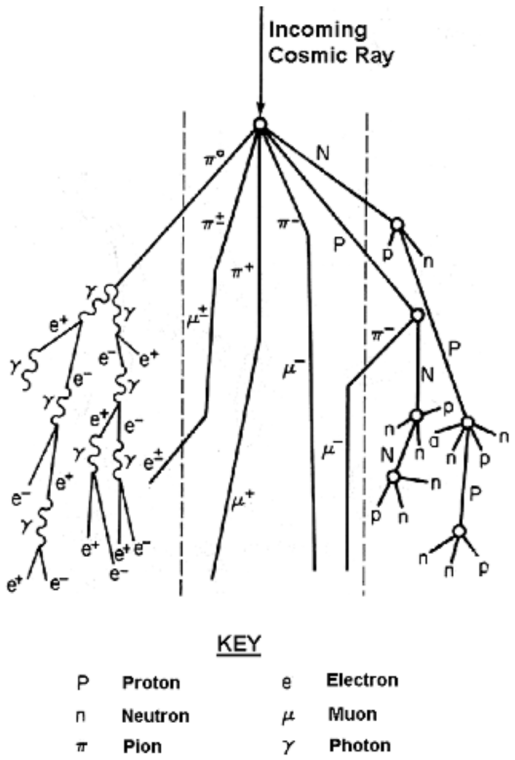


Figure 2.2: Particle cascade (from [10])

2.3 Radiation effects on electronics

The effects of radiation on electronic components are diverse. From long-term non-destructive damage, to instantaneous destruction, the outcomes of an electronic device evolving in a radiative environment are diverse. The study of these phenomena has led the scientific community to categorise them depending on the effect caused by an energetic particle on a component. Nowadays, the distinction is made between instantaneous

damage, and long-term latent damage. Each category implies different phenomena, and the counter-measures differ drastically. These two categories are called respectively single event effects and cumulative effects. An overview of each category is given at the end of this section in figure 2.7.

In addition, it is possible to make a distinction between *non-ionising* and *ionising* radiations. Non-ionising radiations are defined as particles that are not capable of displacing electrons from the crossed materials, while ionising radiations refer to particle with enough energy to remove an electron from its orbit. Therefore, while damages made by non-ionising radiations are often limited to thermal damages, ionising particles can move through substances and alter them as they pass through. Ionising and non-ionising radiations are defined based on their frequency on the electromagnetic spectrum. As displayed in figure 2.3, non-ionising radiation is composed of low frequency radiation, while ionising radiation is composed of high frequency radiation. The separation between these two is around $2,4.10^9$MHz, which corresponds to UV light with a wavelength around 124nm.

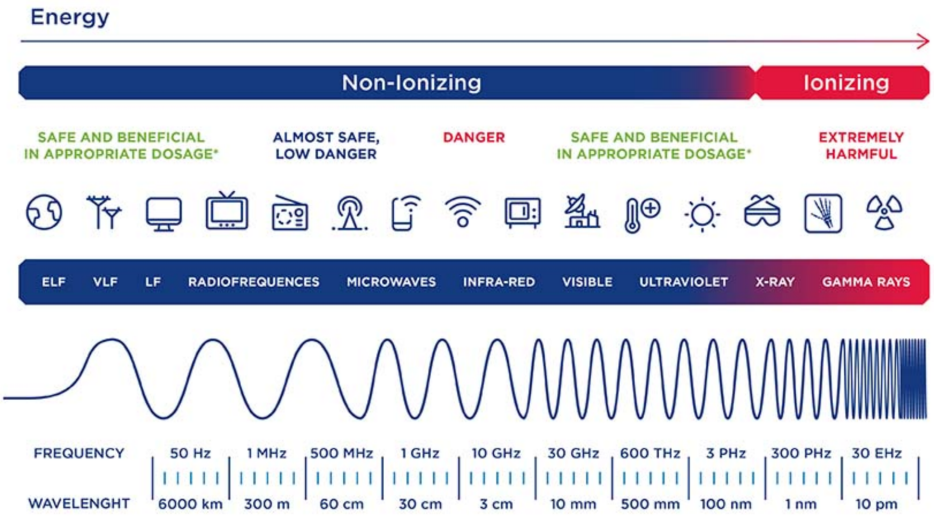


Figure 2.3: Frequency spectrum (from [11])

2.3.1 Single Event Effects

Single Event Effects (SEEs) are faults induced in electronic components by highly energetic particles striking a *sensitive node*. A sensitive node is defined as a node in which electrical potential can be modified by internal injection or collection of electrical charges [12]. A change in the electrical state of a sensitive node can be qualified as a lack of conformity. It becomes a fault when the collected fraction of the charge liberated by an ionising particle is more significant than the electric charge stored on a sensitive node. This type of event is a direct consequence of the radiation phenomena on components. A brief chronology remembering the main steps of the discovery of single event effects is

displayed figure 2.4.

With the invention of *Integrated Circuits (IC)* in 1958 by Jack Kilby, a handful of studies were made in the '60s to evaluate the new possibilities, but also the limitations provided by this new technology. With component size starting to decrease drastically, fears of increased perturbations began to emerge. The possibility of radiation-sensitive components was stipulated in 1962 by Wallmark [13], but this hypothesis was not taken seriously at the time. Nevertheless, with components becoming smaller, it induces that less energy is needed to damage them. Therefore, the probability of being hit by a particle with enough energy to cause such damage increased.
In 1972, anomalies appeared with several satellites. For example, during its mission, the communication with the Hughes satellite was lost for 96 seconds. No explanation was provided for these faults, and investigations started. In 1975, the hypothesis that these anomalies resulted from galactic cosmic rays started to emerge [14]. However, as only four events occurred during seventeen satellite years of operation, plus the fact that the radiation community worked solely with high levels of dose, the idea that a single particle could cause damage gained few supporters.
It is in 1978 that this phenomenon was first described by May [15]. With the possibility of a single energetic particle being able to cause soft errors in electronic components being proved, it is quickly after that the link between galactic cosmic radiation and satellite soft errors was established [16]. Following these discoveries, the hypothesis of hard error caused by space radiation emerged [17].
Later, the distinction between *Single Event Upset (SEU)* and *Single Event Latch-up (SEL)* was made to differentiate between soft and hard errors caused by a single energetic particle. Finally, these faults were categorised as *Single Event Effect (SEE)*, gathering all discovered faults induced by energetic particles. Indeed, even though only SEU and SEL were known at the time, researchers found new phenomena caused by energetic particles.
Regardless of the type, the impact of single event effects on the supply current of a component is often perceived as a *High Current Event (HCE)*.

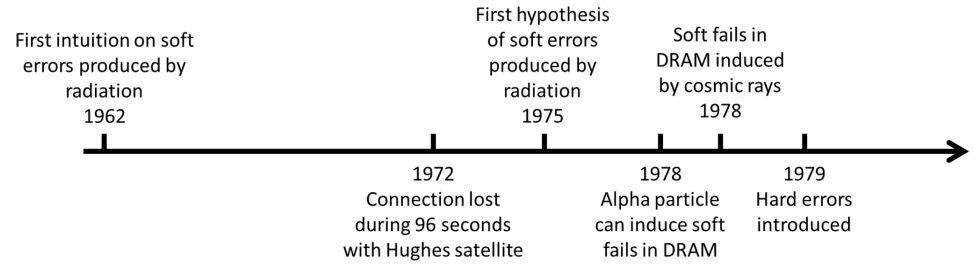


Figure 2.4: Single event effects key dates (dates from [17, 18])

2.3.1.1 Soft errors

Soft errors gather the faults imbued to the signal or data of a component. It is usually the result of memory failure. Thus, it may be corrected by a rewrite of the defective memory cell, a reset, or a power cycling of the component [12].

Single Event Upset (SEU) SEU is a direct modification of a memory cell due to an ionising particle [19]. In figure 2.5, it is shown that SEU is impacting directly the memory by changing the fourth bit. To prevent SEU, an Error-Correcting Code (ECC) can be implemented. SEU happens in micro-electronic devices such as microprocessor, semiconductor memory or power transistors.

Multiple Bit Upset (MBU) MBU is an extension of a single event upset when numerous memory cells are modified simultaneously in the same word. In the case of SEU, a simple ECC can be applied to correct the fault. However, when multiple cells are faulty, a much more elaborated ECC is needed [19].

Single Event Transient (SET) SET is the manifestation of an overcharge generated by an ionising particle on the sensitive node of a transistor [20]. It leads to the creation of a transient voltage that can modify the logical state of a gate, as shown in figure 2.5.

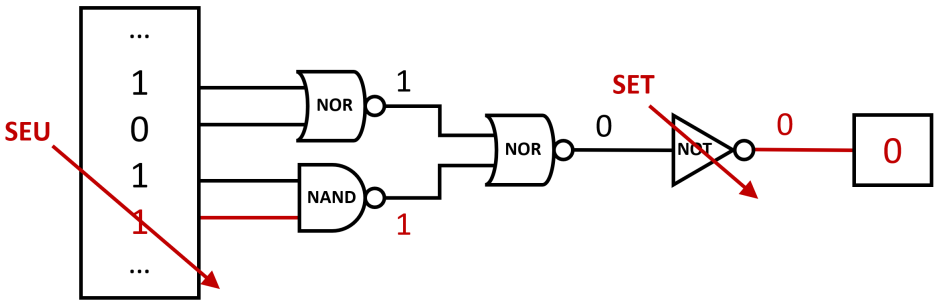


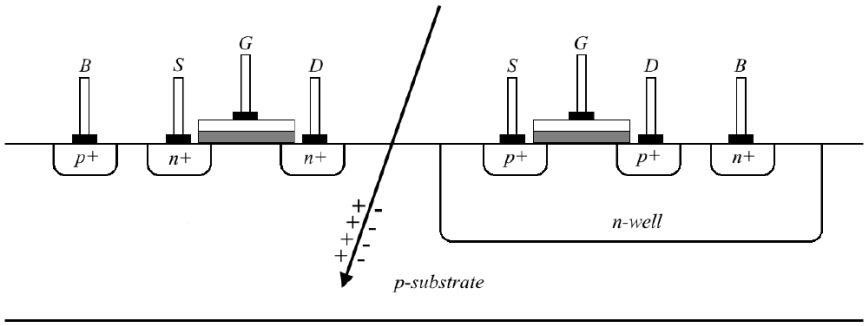
Figure 2.5: SEU and SET examples on a circuit

Single Event Functional Interrupt (SEFI) SEFI corresponds to a temporary interruption of the functionality of complex devices. Its origin can vary and lasts as long as a power cycle is not performed [21].

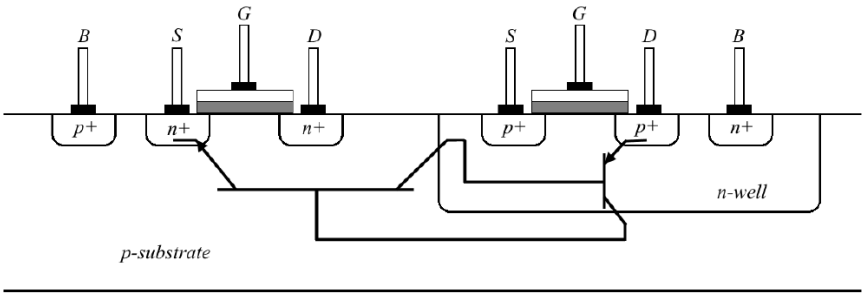
2.3.1.2 Hard errors

Hard errors are non-recoverable issues and cause permanent degradation of the component [22]. They can also be referred as *destructive anomalies*.

Single Event Latch-up (SEL) A single event latch-up is a phenomenon caused by an inherent parasitic structure in the CMOS technology. it can be modelled as two bipolar transistors, NPN and PNP and multiple resistances representing a specific substrate region [23]. When an energetic particle strikes through the substrate between the transistors, it creates electron-hole pairs allowing charges to travel throughout different regions of the substrate. If the moving energy is high enough, a low impedance, high current path starts a short circuit between the transistors, as shown in figure 2.6. The resulting heat generated can damage the component. Recently, the term



(a) Energetic particle colliding with the substrate



(b) Resulting short circuit

Figure 2.6: Single event latch-up modeling (from [24])

micro latch-up started to emerge [24]. It defines a single event latch-up that is not damaging enough to immediately destroy the component. It is characterised by a discrete step in the supply current. It results in the deactivation of some functions and an additional strain on the component. Moreover, micro latch-ups can add up in different component locations, creating significant damage.

Single Event Snapback (SESB) A single event snapback occurs when a high current is found between the source and the drain of a single NMOS transistor. Then, through the action of an ionising particle, a parasitic NPN bipolar transistor between the source and drain is activated, resulting in a local overheating of the component.

Single Event Hard Errors (SEHE) Single event hard errors are very similar to single event upset, with the exception that the damage done to the memory cell is permanent.

Single Event Gate Rupture (SEGR) Single event gate ruptures results from a heavy ion striking the drain region of a power MOSFET [25]. It manifests by an increased gate leakage current that can lead to the complete failure of the device.

Single Event Burnout (SEB) Single event burnout is primarily observed in power bipolar transistors and MOSFETs. It is the result of a heavy ion causing triggering the component combined with the avalanche effect. The component suffers from thermal degradation that can lead to the complete failure of the device [22].

2.3.2 Cumulative effects

As opposed to single event effects that are probabilistic one-shot damage, cumulative effects increase with continued exposure to radiation. They can be compared to ageing effects as the component characteristics gradually degrade through its lifetime. Also, it is important to note that cumulative effects are permanent, and the component cannot be restored by a power reset.

2.3.2.1 Total Ionizing Dose (TID)

Total ionising dose corresponds to the cumulative effects caused by ionising particles over an exposition time. Indeed, charged particles cause an electrostatic force on a material that they cross. The result is the creation of electron-hole pairs due to exited electrons shifting from their bound state. In short, it corresponds to the cumulative energy absorbed by the component. The unit used is *rad* (Radiation Absorbed Dose). TID effects on a component include leakage current, threshold voltage, functional failures...

To reduce TID effects, *device shielding* is commonly used for space missions. This protection method is going to be covered in section 2.5.1.

2.3.2.2 Displacement Damage Dose (DDD)

Displacement damage dose has similar long-term effects on a component to TID. However, the mechanism is different. DDD results from the displacement of multiple nuclei from their lattice position through time. If too many nuclei are altered, then the component's material property are altered.

2.4 SEE testing methods

Being able to evaluate the radiation levels a component is going through during the whole time of a space mission is essential to implement efficient counter-measures.

Therefore, ways to simulate radiation faults on Earth are of prime interest to space agencies, and many techniques are available today. However, as we saw, a wide variety of radiation faults exists, and choosing the right test is never an easy task, as many parameters have to be taken into account.

Nevertheless, when working in anomaly detection, the main obstacle is usually the accessibility of faulty data. Indeed, it can be complex to gather representative data of all anomalous behaviour of a system. In case of space applications, single event effects are miscellaneous, thus, it is almost impossible to get all possible appearance

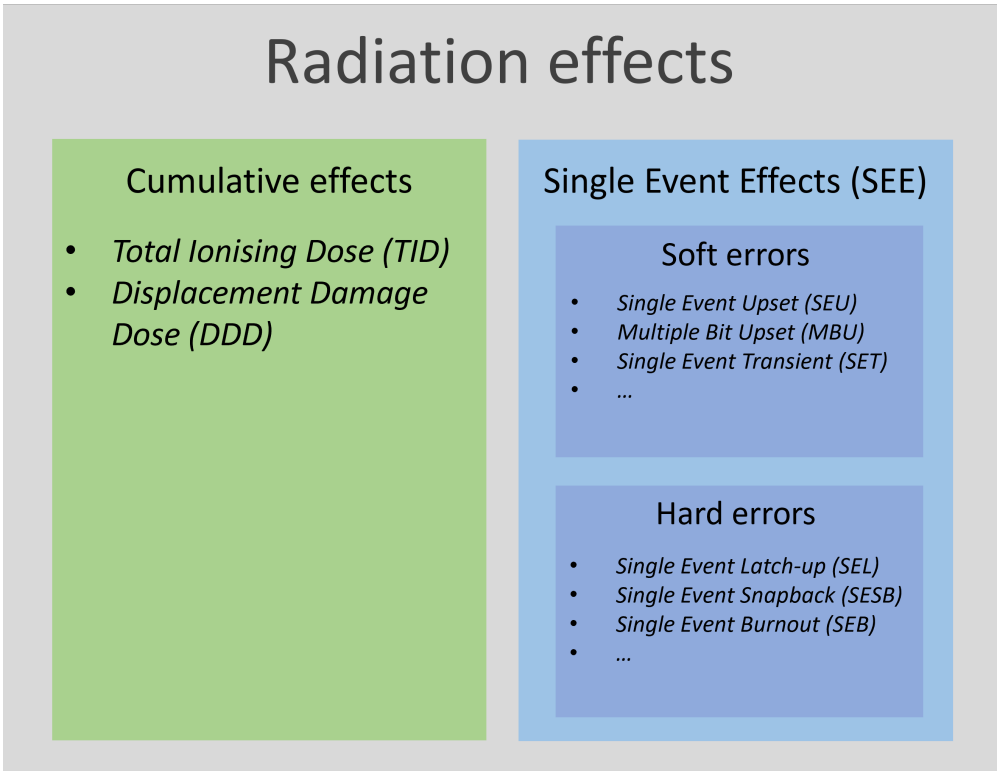


Figure 2.7: Radiation effects insight

for a specific device. To do so, it would require to test each sensitive node of a component.

Another solution is to evaluate the number of SEEs during a mission. For this, it is possible to evaluate the amount of energy perceived by a material using the linear energy transfer value. From there, it is possible to calculate the cross section of a component to evaluate the number of SEEs expected during a mission.
Then, different testing methods exist to simulate to simulate single event effects on a specific component

2.4.1 Linear Energy Transfer (LET)

The Linear Energy Transfer (LET) corresponds to the amount of energy transferred to a material traversed by an ionising particle per unit of distance. It is influenced by the particle's nature and the traversed material. The LET is often used as a unit of measure during experimental testing, or to design space missions. Therefore, it is a precious indicator to predict the possible damage that can be done to a specific component.
It is defined in Eq. (2.1). Its unit is MeV/cm. Still, because the energy loss is proportional to the density of the traversed material, it is possible to express the LET divided by the material density. In that case, its unit is $MeV.cm^2.mg^{-1}$

$$LET = -\frac{1}{\rho}\frac{dE}{dx} \tag{2.1}$$

with ρ the density of the material, dE the energy loss of the charged particle and dx the distance travelled by it in the material.

2.4.2 Cross section calculation

The cross-section corresponds to the measurement of the sensitive area of a component as a function of energy [24]. It indicates the probability of an event involving an ionising particle at a given energy. Using the cross-section, it is possible to estimate the number of single event effects happening during a space mission and thus, design suitable protection. The cross-section can be measured by counting the number of events triggered on a component and comparing it to the flow of particles irradiating the component.
The aim goal is to create a cross-section curve for various LET values. From there, it is possible to distinguish two key values. The first one is the LET threshold level at which SEEs start to appear. The second one is called the saturation threshold at which the number of SEEs is stabilising (see figure 2.8).
The cross-section is calculated using Eq. (2.2) and its unit is cm^2.

$$\sigma = \frac{N_{SEE}}{\phi} \tag{2.2}$$

with N_{SEE} the number of events recorded and ϕ the particle fluence.

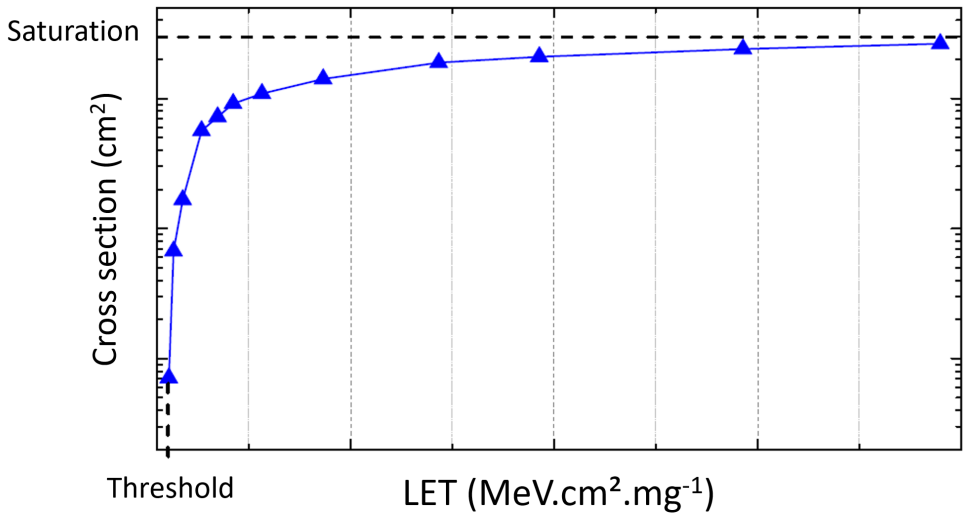


Figure 2.8: Cross section as a function of LET (from [26])

When calculating the cross-section of a component, it is possible to predict its durability when exposed to a certain quantity of radiation. Doing so makes it possible to choose the most suitable component for a space mission.

2.4.3 Californium-252

To emulate a radiative environment on Earth, a Californium-252 (Cf252) radiation source can be used. Californium-252 sources emit various types of radiation. The list goes from alpha particles, beta particles, gamma particles, neutrons and around 3% of heavy ions. It is why Cf252 is an adequate source to be used to test single even effects on a component. The main inconvenience is that its emission radius is short. With only 15μm maximum, it can fail to reach the deepest part of a component [27, 28].

2.4.4 Laser

Laser testing for SEE is first introduced in 1965 [29]. At the time, only X-rays or particle generator testing is available to emulate the effects of radiation on a component. Laser is only seen as an inexpensive, yet powerful tool during a system design's phase, but is not able to replace entirely traditional testing methods. Limited at the time to single event transient, studies started to emerge using a laser in a large spectrum of single event effects [30–32].
The main inconvenience for IC is the metallisation layers located at the top of the components. These layers can prevent the laser beam from accessing critical nodes on the chip. That is why that backside testing started to emerge in the 2000s [33]. Nowadays, laser testing is gaining much interest due to its ease of access. It is seen during the whole design process of a system, and is often used as a pre-characterisation phase before heavy-ion testing.

2.4.5 Heavy ion testing

Heavy ion accelerators are considered the standard procedure when testing for single event effects. When using this method, particles are accelerated by using an electric field [27, 30]. The resulting beam is directed on the device under test previously placed in a vacuum.

In general, two types of accelerators can be used for single event effect testing. First, linear electrostatic accelerators uses two electrodes to accelerate ions. This type of accelerator is using the resulting acceleration from a change state of ions provoked by the electrodes. The other common type is called cyclotron. This type of circular accelerator uses an electric field in combination of a curve trajectory induced by a magnetic field to accelerate ions.

2.5 Protection methods

2.5.1 Shielding

Shielding is mainly used to prevent TID effects. The sensitive component is surrounded by a protective material. Aluminium is a common shielding material as it can effectively stop electrons. Also, the protection design can be tedious, as numerous parameters, such as material composition, thickness, and geometry have to be taken into consideration [34].

2.5.2 Radiation-hardening

Radiation-hardening is the concept of improving the protection of a component against its radiative environment. Usually, hardening in the space industry refers to a modification of the manufacturing process of a specific component. Many techniques already exist, such as increasing the distance between transistors to avoid single event latch-up or adding a decoupling capacitor at the gates of a device against single event transient. Unfortunately, these techniques add a lot of complexity to a component, as they have to be designed for a specific device, thus adding development cost. As a result, hardened components are drastically more expensive than their COTS counterparts.

It is why the space industry is looking more and more to new means of detecting radiation faults, that are not components dependent.

2.5.3 Anti-latch-up system

An anti-latch-up system is a part of what is called *system-level latch-up counter-measures* [24]. It is the method currently used for most space missions. The concept is fairly simple. The supply current of the component is monitored to find any high deviation that would be diagnosed as a single event latch-up. In other words, it is a threshold-based method. This method can be implemented in various ways and is a low-cost solution. However, even though this method is efficient in detecting destructive single event effects that heavily influence the supply current, it cannot detect minor faults that could still lead to long-term damages. Indeed, because of the TID effect and the deviation resulting in the supply current, the threshold value must be taken with a high margin. It results in an important non-detection zone.

In this work, the anti-latch-up system is considered as a reference in the detection of single event effects. It is referenced as the *baseline threshold detection method*.

2.5.4 NOSTRADAMUS

NOSTRADAMUS is a detection method developed by CNES in order to improve the quality of spacecraft monitoring. This method is based on a machine learning algorithm, and more specifically, a combination of Principal Component Analysis (PCA) [35] and One-Class Support Vector Machines (OCSVM) [36]. The main drawback found during the implementation of this project was the high number of false alarms during testing [37].

2.5.5 Latch-up Detection and Protection (LDAP)

LDAP is a protection chip developed by Zero-Error Sytems (ZES). This solution is a three-stage detection method that first focuses on absolute current values(i). Then, the second stage is configured to detect the rate of change of supply current (di/dt). Finally, the power reset decision is taken by evaluating the previous two values with a fixed threshold [38]. The advantage of this method is that the detection is performed by an independent chip. Therefore, it is possible to use every component available, including COTS.
Airbus provided 2.5 million dollars to ZES in order to scale its method [39].

2.5.6 Adaptive threshold

INVAP recently fielded a patent for a new single event latch-up detection system [40]. It can be seen as an improvement of the classic threshold detection method that takes into consideration the TID effect. As stated before the TID effect is responsible for a deviation of component characteristics. The idea proposed by INVAP is an adaptive threshold that is able to follow the deviation. By doing so, it is possible to set up the threshold closer to the nominal behaviour of the component. Figure 2.9 shows an example of such a method. The supply current monitored labelled as "801" clearly display a deviation of its behaviour. In this example, the threshold labelled "800" is able to follow the deviation, allowing a much closer and more precise threshold than without applying this adaptive method.

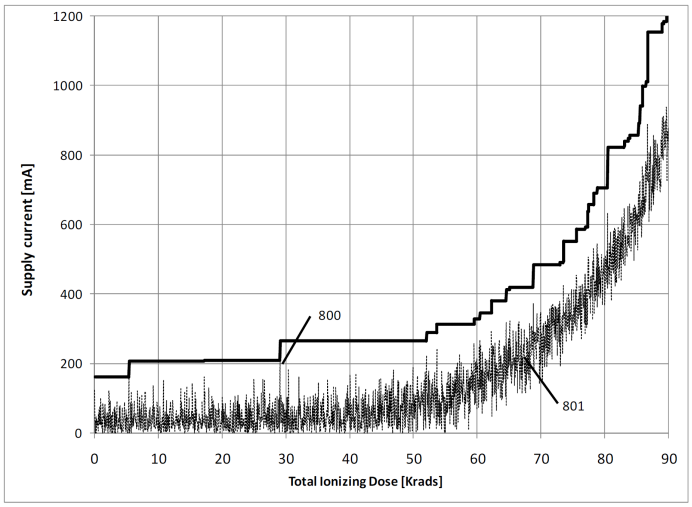


Figure 2.9: Adaptative threshold (from Cibils [40])

2.6 Conclusion

In this chapter, a detailed overview of the radiative environment that space components undergo is provided. Diverse sources are responsible for the emission of energetic particles. These particles are the cause of various damages to electronic components that range from long-term damages to instantaneous destruction of devices.
To counter radiation faults, extensive testing has been performed to improve the understanding of single event effects. By doing so, it has been possible to design protections suitable for specific components or faults. Nevertheless, today's baseline method fails in detecting tiny single event effects. New approaches aim to improve the detection of such faults (LDAP, adaptive threshold) and many resources are deployed to enhance the durability of space components.

In that context, anomaly detection using machine learning techniques is a stimulating domain to investigate.

Chapter 3

Machine learning for anomaly detection

Contents

Artificial intelligence is a computer science field that aims to build machines capable of mimicking intelligent behaviour. Indeed, even though machines are able to memorise and manipulate tremendous values that are way beyond human comprehension, some basic tasks still remain impossible to be performed by a computer. For example, comparing two cat pictures, describing a scene, having a simple conversation about the weather or composing a catchy melody might seem trivial for the reader, but reveals to be highly complex for our binary friends. Artificial intelligence aims to decrease the gap between machines and humans in these tasks.

Artificial intelligence bases its approach on the concept of *agent*. An agent is anything that is able to perceive its environment using sensors and then acts upon this environment using actuators [41].
From there, artificial intelligence can be divided into two main fields. The first one is *symbolic AI*. The most commonly cited method of symbolic AI is *expert systems method* [42]. These require knowledge of the system.

The focus of this thesis is on the second field, called *machine learning* [43]. This area focuses on improving the agent's performance through multiple observations of its environment. Machine learning usually works in two phases. First, a *training phase* is performed in order to model the system. In this phase, the machine learning algorithm tries to create relations between the given inputs and outputs. Then the *prediction*

phase uses the previously created model to adapt to new and unknown input data. The advantage of machine learning is that it can adapt and generalise to situations that were not considered by the programmer. Moreover, it is powerful in cases where the system is too complex to be programmed manually (as for facial recognition or self-driving car).

Anomaly detection is one of the use cases of machine learning [44]. Anomaly detection finds its interest in a wide variety of applications. From fraud detection, cyber-security, to health care and surveillance, anomaly detection is of the utmost importance. In the case of space applications, anomaly detection techniques can be introduced to improve the detection of single event effects, by finding anomalous patterns that deviate from the normal behaviour of space components.

In this chapter, the focus is on anomaly detection. It resolves around two questions:

- *How to characterise single event effects as anomalies?*
- *Which machine learning methods could be used to improve single event effects detection?*

First, the different types of anomalies are described in section 3.2. Three types of anomalies are mentioned, as well as a discussion about the definition of single event effects. Then, the three broad categories of anomaly detection techniques are detailed in section 3.3. In these categories are listed some of the most common anomaly detection methods. Finally, some of the most encountered distance metrics used in these methods are listed in 3.4.
Before diving into technical considerations, a run-through of some key dates of artificial intelligence is proposed in section 3.1. It is done in chronological order, and some of the methods described will be detailed in the following sections.

3.1 A brief history of artificial intelligence

Artificial intelligence has known an incredible gain of interest since the beginning of the 21^{st} century, mainly in the machine learning sub-field. Firstly for its performance in image recognition, it quickly spread to almost all engineering fields, and became the focus of many societal issues. However, artificial intelligence, is in fact, an old field of research that has met a dead end two times already. Referred today as *artificial intelligence winter*, these events are characterised by a drastic disinterest in the field for an extended period.

It is possible to trace the foundation of artificial intelligence back to ancient times with philosophers like Aristotle or Descartes, or with major mathematical foundations like formal logic or probability. However, in this brief history of artificial intelligence, the focus is on modern interpretations of artificial intelligence.

Artificial intelligence started with the goal of mimicking the human brain's functionalities. It is possible to pinpoint its beginning in 1943 with the first definition of the artificial

neuron [45]. Published in the bulletin of mathematical biophysics by McCulloch and Pitts, this publication shows the strong link that artificial intelligence always had with biology. Indeed, this preliminary work was to give a better understanding of the complex mechanisms of the brain to achieve intelligent behaviour.

This work was completed in 1949 by Hebb, which stated the rules in which the neurons behave and are able to learn together. It is then that the term *connectionism* started to emerge. Often summarised as *"cells that fire together, wire together"*, these rules describe the plasticity of the brain cells by pointing out the efficiency of continuous and repeated activity on two synaptic cells.

A year later, a groundbreaking article is published. In 1950, Turing stated the possibility of intelligent machines [46]. Even though, from the author's consent, no concrete proofs are given, this article is one of the foundations of the artificial intelligence field. Moreover, it nourished many science fiction works, such as *2001: Space Odyssey by S.Kubrick (1968)*, or more recently *The Turing Test by Bulkhead Interactive (2016)* or *Detroit: Become Human by Quantic Dreams (2018)*.

In 1952, Samuel writes about what we consider today as the first machine learning example. An algorithm is being trained onto numerous games of checker. By each game played, the algorithm learns and improves its performance [47]. The ability to "foresee" future moves is performed using a *tree of moves*. From a given position, the algorithm chooses the optimal move based on its previous experiences recorded in this tree. By doing so, the machine is able to perform better than an "average novice".

It is in 1958, following the footsteps of its predecessor, that the *perceptron* is defined by Rosenblatt [48]. Thought as a binary classifier, this algorithm uses first-hand examples to modify the weights of an artificial neuron, resulting in the linear separation of two classes. The ancestor of modern neural networks is born. Figure 3.1 display a scheme of Rosenblatt's perceptron. On the left part, each input x_i is weighted by a coefficient w_i. The output y is given by the result of an activation function f on the total sum of each weighted input. Traditionally it is possible to add a bias β to the total sum.

In 1967, the first formal properties of the *k-nearest neighbour* algorithm (KNN) are formalised [49]. Even though it is possible to trace the first introduction of such classification method in 1951 by Fix and Hodges, 1967 marks the start of a long investigation to improve the nearest neighbour algorithm to what we know nowadays. KNN is one of the most popular classification methods and is still widely used by industrials alongside academicians.

After that, artificial intelligence interest decreased drastically. This period between 1969 and 1982 is known as the *first AI winter*. With high expectations crushed by many failures, fundings were stopped. Plus, research on connectionism took a critical strike when Minsky and Papert described the limitations of the perceptron in 1969 [50]. One of the main arguments states that the perceptron is unable to achieve some very simple

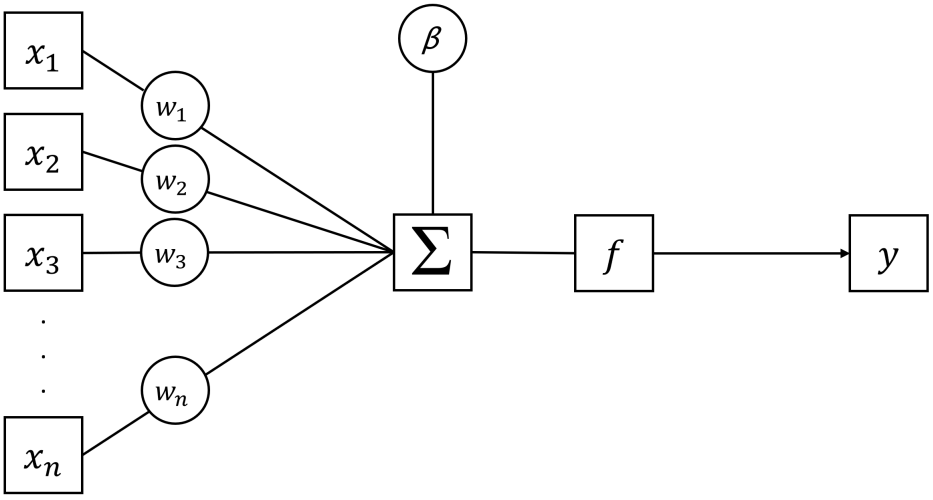


Figure 3.1: Rosenblatt's perceptron

problems. Indeed, it was proven that the perceptron could not apprehend the logical function XOR, because of the non-linear nature of the problem.
In addition, technical limitations started to arise. As computers were not powerful enough to run complex neural networks, many of the promises made by this field were out of reach at the time. Indeed, first breakthroughs were performed on problems with few objects, and scaling proved to be more difficult than AI researchers encompassed .

During this period, efforts are focused on another aspect of artificial intelligence: symbolic AI. Expert systems mainly represented this sub-field of AI. Rather than relying on specific formalisms and inference schemes like connectionism, expert systems focus on domain knowledge to allow larger reasoning. It can be broken down into a set of logical rules dictated by the programmer that the program has to follow to give its prediction. Even though it is limited by the knowledge of the expert community to resolve a problem, expert systems gave more consistent and deterministic results. One of the first examples of an expert systems program is the DENDRAL program created by Buchaman in 1969. Buchaman teamed up with chemical experts in order to describe molecular structures corresponding to input mass spectrum and formula of an organic chemical compound [51]. The important milestone of this project is that Buchaman proved the feasibility of DENDRAL, allowing it to be seen as a reliable method for real applications.
From there, *knowledge-based systems* became more and more popular. The massive project called the Fifth Generation Computer Systems (FGCS) funded in Japan in 1981 had a major impact on promoting knowledge-based systems worldwide. This project aimed to create powerful computers able to use massively parallel computing and logic programming.

The regain of interest for AI started in 1982, due to new technological breakthroughs that went to invalidate Minsky and Papert's propositions. When they restricted their argumentation to Rosenblatt's perceptron, they did not emphasise that the improvement

of learning rules would allow multiple layered architectures.

The next few years are marked by the publication of the most well-known neural networks used today:
First, the *Self Organizing Maps (SOM)* by Konhonen [52] is published in 1982. This architecture enables the mapping of high dimensional in 2D spaces using a topographic neuron model that can influence other neurons in proximity.

A milestone for artificial intelligence happens in 1985 with the work of LeCun and Parker on the backpropagation of neural networks. Backpropagation consists in propagating the error through the different layers of a neural network during the training phase. Even though they were not the first to use the backpropagation method, they demonstrated that an efficient backpropagation was possible for neural network applications.
In 1986, Hinton developed a new architecture relying on input and hidden neurons called *Boltzmann Machines (BM)* [53]. This specificity gives the ability to understand unknown or complex information about the system.

Next, in 1988, a well-known architecture called *Auto-Encoders (AE)* is published by Bourlard [54]. This architecture is similar to classic neural networks architecture but with the output being identical to the input. From there, it is possible to use auto-encoders in many ways (feature extraction, encryption, feature reduction, de-noising...).

In 1989, LeCun et al. published a groundbreaking article in which *Convolutional Neural Networks (CNNs)* are used to recognise hand-written numbers successfully [55].

In the same year, a new axis of artificial intelligence emerged from the thesis of Watkins [56]. With its proposal of what is now called Q-Learning, reinforcement learning was born. It varies largely from the machine learning methods discovered until then, as no input nor outputs are explicitly given to the algorithm. Instead, an agent evolves on its own in an environment by performing actions. The training is based on rewards given to the agent depending on the output of its actions.

In 1990, a second winter occurred. Similarly to the first winter, artificial intelligence did not fulfil the promises made during the last decade, and its popularity decreased. At the time, computational power is still insufficient to run complex neural networks.
Moreover, some argue that the cause of this second fall started way back in the 70s: The game of change in the command chain of the *Defense Advanced Research Projects Agency (DARPA)* resulted in significant cuts in the funding in AI. It is interesting to note that those cuts did not happen because of hostility in regards to artificial intelligence fields, but instead because DARPA estimated that the funds allowed to AI were disproportionate in regards to other scientific fields, such as supercomputing [57]. Moreover, AI researchers failing to see that the funding cuts were primarily responsible for the lack of new projects, started to divide themselves. The connectionism experts started to label expert systems as "not really AI" and pointed out the flaws of such methods. Therefore began a quarrel that is continued today.

Some significant works are still published during this period. In 1995, Cortes continued a work started by Fisher in 1936 on a binary classification method using couples of vectors [58] [59]. Called at the time support-vector-networks, these are known today as *Support Vectors Machine (SVM)* and is still one of the most used classification methods nowadays.
The same year, Ho wrote about a new method that uses multiple decision trees to improve prediction results. This algorithm is called *Random forests (RF)* [60].

It can be complex to pinpoint precisely the end of the second winter, as multiple references vary from 1997 to 2001. However, the defeat in May 1997 of the world chess champion Garry Kasparov to Deep Blue, the supercomputer of IBM, is a significant event in artificial intelligence. When it was thought impossible for a machine to win against a pro player, the defeat of the world champion shifts entirely the opinion of the public eye towards AI.

In 1999, Nvidia released, as they called it, *the world's first Graphics Processing Unit (GPU)*: the GeForce 256. The parallelisation possibilities offered by GPU and the massive jump in computer performance finally allowed to run complex neural networks models efficiently.
At the beginning of the 21^{th} century, the advent of internet led to a massive increase in data collected. Extensive and labelled databases of various kinds started the era of what is referred today as *big data*. Hand analysis, which was sufficient in the past, quickly became too complex, and new ways of treating information were needed.
With the new popularity of AI, the technological breakthrough of GPUs, plus the availability of large amounts of data, artificial intelligence became the centre of attention. Following these breakthroughs, a great number of projects involving AI flourished from 2001 to 2009. Then, significant successes started to arise.

2009 marked the grand entrance of neural networks in computer vision competitions with the victory of Jürgen Schmidhuber in the ICDAR French Connected Hand-written Competition using a fast deep neural net. Since then, neural networks have won almost all competitions they entered in, being image segmentation, object detection or object classification, with accuracy improvement of up to 40% from other technics.

In 2010, Google announced its self-driving car system that was able to drive more than 225 000 km without any accidents. Even though the field of self-driving cars is not novel, with Tsukuba's lab that demonstrated the first autonomous car able to follow a line at 30km/h in 1977, it is only since these announcements that funding started to grow in this field.

In 2013, The Facebook AI Research (FAIR) group is created in order to improve state-of-the-art AI. Their work benefits greatly facial recognition field but also the generative image field with major work in self-supervised learning. Notably, they worked on scaling Generative Adversarial Network (GAN), a network model resolving into two competing networks proposed by Goodfellow in 2014 [61].

In 2018, Nvidia shakes the computer graphic field by realising their RTX series. Besides a totally new Turing architecture enabling real-time ray tracing, RTX cards are equipped with *Deep Learning Super Sampling (DLSS)*. This technology uses a convolutional auto-encoder neural network to enhance and upscale low-resolution images to be rendered in real-time at higher resolution. Some benchmarks showed double in performance when this feature is enabled.

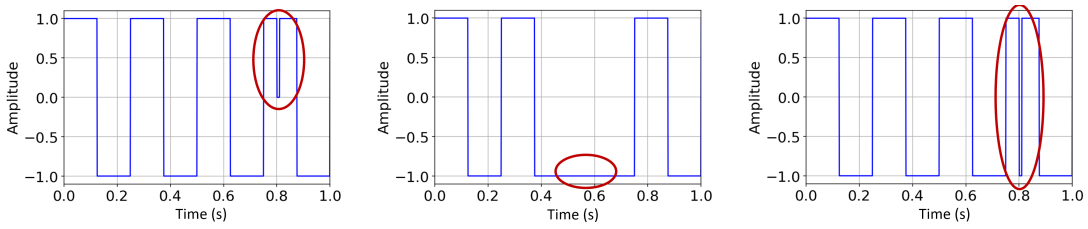
Nowadays, Artificial intelligence has grown in almost all industrial fields. From anomaly detection in telemetry [62], to new diagnosis methods in medicine [63, 64], control simulation [65] or audience recommendation [66, 67], artificial intelligence is everywhere.

With the recent expansion of machine learning at the beginning of this century, many think that it is a new field of research, whereas most of its methods have been created during the last century. Moreover, it is interesting to realise all the setbacks that machine learning went through to see in a new light the recent success of this field.

From there, the next sections are going to focus to the specific field of machine learning of anomaly detection.

3.2 Anomaly detection

Anomaly detection refers to the problem of identifying observations that deviate from what is defined as the system's normal behaviour. These observations are called anomalies or outliers. It is possible to discriminate between three types of anomalies [44]. These are point anomalies, collective anomalies and contextual anomalies.



(a) Point anomaly **(b)** Collective anomaly **(c)** Contextual anomaly

Figure 3.2: Anomaly types

3.2.1 Point anomalies

A single observation that deviates from normal behaviour is considered a point anomaly. It is the simplest type of anomaly. It is displayed in figure 3.2a. In this figure, the observation at 0.8s takes a value that is impossible to encounter when the system runs normally.

3.2.2 Collective anomalies

When multiple observations taken together deviate from normal behaviour, it is considered as a collective anomaly. In figure 3.2b, if taken individually, each observation inside the red circle is already observed, thus they are valid values for this data set. However, it is the combination of all these observations that is problematic. Indeed, the lower state lasting between 0.4s and 0.7s is twice as long as it should be. It is a sign of anomalous behaviour.
Another example would be when considering multiple data sets altogether. For example, let us take the example of a hospital with 100 patients. The average heart rate is around 80 beats per minute (BPM), with a normal state between 60 BPM to 100 BPM. A point anomaly would be that a single patient having a heart attack will see heart rate going up to 150 BPM. However, if all patient's heart rates rise to 100 BPM, even though it is still at the acceptance rate and not considered a point anomaly, the average cannot be considered normal behaviour. It is then considered a collective anomaly.

3.2.3 Contextual anomalies

When an observation differs from the anticipated scenario, it is considered as a contextual anomaly. The border between contextual anomalies and the two categories described earlier is difficult to establish. Indeed, both point and collective anomalies can be considered as contextual anomalies. To be able to distinguish contextual anomalies, a priori knowledge of the system behaviour is required. For example, in figure 3.2c, the observations made at t=0.8s have the same value as any observation of the lower cycle. However, it is the context of the data set that gives the information of an anomaly. As the steady state at t=0.8s is the higher cycle, the normal value should be equal to the observations of the higher cycle.

3.2.4 Single event effects as anomalies

The topic of this thesis project is to improve single event effects detection. As referenced in section 2.3.1, the impact of a single event effect on the supply current is perceived as a high current event. It corresponds to a persistent shift in the supply current. From there, it is possible to categorise single event effects based on the types of anomalies. For this, a distinction is made between destructive and non-destructive single event effects.

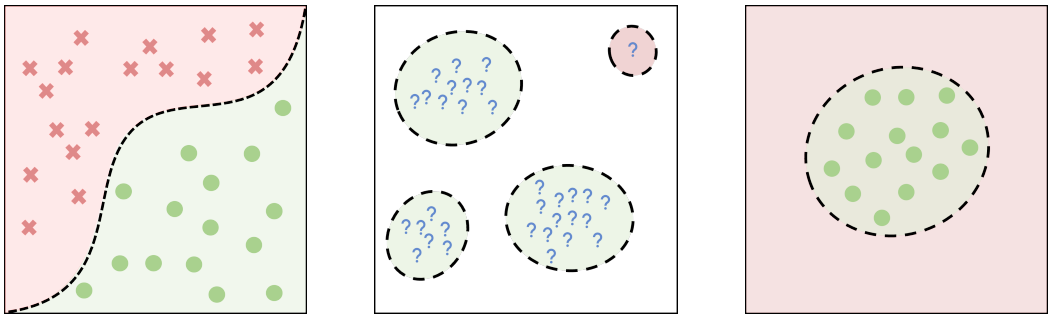
Destructive single event effects result in a significant shift of the supply current. The resulting values are heavily deviated from the normal behaviour. Therefore, even though multiple faulty observations are resulting from single event effects, it is possible to define them as point anomalies, as each individual observation differs from the normal behaviour.

On the other hand, the supply current deviation resulting from a non-destructive single event effect can stay hidden in the normal behaviour to the naked eye. Therefore, it is not possible to characterise these anomalies by looking at a single point, but multiple observations have to be taken into consideration. Consequently, it is possible to characterise these non-destructive SEEs as collective anomalies.

3.3 Categories of anomaly detection methods

Supervised learning, semi-supervised learning and non-supervised learning are the three main categories of learning that govern anomaly detection. Based on the problem to solve, one might use one of each category.

Anomaly detection depends on the availability of *labels* that characterise observations as normal or anomalous. Anomaly detection models use these labels to extrapolate a model of the system during the training phase. Then during the prediction phase, the model is used to predict new observations as normal or anomalous. It is the availability of these labels that defines the category of learning method used.
Note that in general machine learning, these labels represent the target information of an observation.



(a) Supervised learning **(b)** Unsupervised learning **(c)** Semi-supervised learning

Figure 3.3: Anomaly detection categories

In addition, a fourth category exists in general machine learning. Called reinforcement learning, it is a particular case of machine learning where no data has to be provided during the training phase. Reinforcement learning is not used in anomaly detection. However, the author finds interesting to detail the possibilities of reinforcement learning. Therefore, it is possible to find a description of reinforcement learning in section A in the appendices.

3.3.1 Supervised learning

In machine learning, the learning task is about modelling a system based on input-output pairs. The particularity of supervised learning is that the nature, or label, of each observation contained in the training set is given to the algorithm. An observation is composed of *features* representing measurable characteristics of the system. Formally, the training stage can be written as follows:
Given a training set X of N observations as

$$X = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$$

Table 3.1: Confusion matrix

		True values: Positive (P)	True values: Negative (N)	
Predicted values	Positive (PP)	True Positive (TP)	False Positive (FP)	Precision (Positive Predictive Values) *PPV = TP/(TP+FP)*
	Negative (PN)	False Negative (FN)	True Negative (TN)	Negative Predictive Values *NPV = TN/(FN+TN)*
		Sensitivity (True Positive Rate) *TPR = TP/(TP+FN)*	Specificity (True Negative Rate) *TNR = TN/(TN+FP)*	Accuracy *ACC = (TP+TN)/(P+N)*

with x the input feature vector and y the output value and being related by a function such that $y = f(x)$, find a function h that approximates the true function f [41].

In general machine learning, it is possible to distinguish between regression and classification algorithms. A *regression algorithm* corresponds to the case when the output y is a quantity. The model aims to understand the relationship between independent (y) and dependent (x) variables. Therefore, it can be used to understand the relations between variables. However, its main applications dwell in the forecast of a system [68]. The most common regression method is linear regression and logistic regression. Regression is rarely used in anomaly detection, but some research can be found on the topic [69, 70].

On the other hand, *classification* algorithms are the most used for anomaly detection. Classification corresponds to the task of being able to differentiate objects. For instance, differentiating between the picture of a cat and a dog is one of the most used example. Classification is when y corresponds to specific labels or categories. The goal is to propose a model that can identify different categories, or classes, based on their inputs. The classification task can be extended to multi-class classification, where the algorithm tries to separate multiple known classes. The label for each sample is known, and all classes are represented, as in figure 3.4.
In anomaly detection, supervised learning can be considered as a classification task with only two classes: normal and anomalous, as shown in figure 3.3a

A critical aspect of classification is to judge the performance of the classification model. Multiple evaluation factors can be used to assess if the modelled function h is satisfactory. Most of them rely on the *confusion matrix*, as displayed in table 3.1. It is a powerful tool that allows quick visualisation of the performances of a model for a specific data set.
It primarily reports all possible prediction outcomes for true and predicted class:

- **True positive:** Positive sample predicted positive
- **True negative:** Negative sample predicted negative
- **False positive:** Negative sample predicted positive
- **False negative:** Positive sample predicted Negative

From there, multiple indicators can be calculated. As each of them delivers specific information on the model performances, it is essential to cross-validate the results using various indicators.
The most common are explained below:

- **Sensitivity or recall** refers to the probability of correctly predict positive classes.
- **Specificity** refers to the probability of correctly predict negative classes.
- **Precision** refers to the probability of being correct when predicting a positive value.
- **Negative predictive value** refers to the probability of being correct when predicting a negative value.
- **Accuracy** refers to the overall probability of the model being correct on each prediction.

These indicators are precious to evaluate the performance of a classification model.

As it appeared in section 3.1, many algorithms exist in order to perform similar tasks. Classification algorithms are no different, and it is essential to understand the advantages and drawbacks offered by each technique when trying to model a system. The following are examples of the most common classification algorithms. Note that some of these algorithms, such as neural networks, can be used for classification and regression.

- *K-Nearest Neighbors (k-NN):* k-NN is a non-parametric method that consists in finding the class of a point based on its k nearest neighbours in the feature space [71].
- *Naïve Bayes:* This method is based on the Bayes's theorem defined in equation 3.1 with a strong independence assumption between the features.

$$P(A\backslash B) = \frac{P(B\backslash A).P(A)}{P(B)} \tag{3.1}$$

- *Decision Trees:* This method uses tree-like models [72] where the target variable can take a value in a discrete set. In these structures, the leaves represent class labels, and branches represent conjunctions of feature values. This method is prevalent because of its easy-to-understand principle.

- *Random Forests*: This method is an *ensemble learning* method, meaning that it is based on multiple agents working together to get a better result. For the random forest algorithm, a multitude of decision trees is used to build the prediction [60].

- *Support Vector Machines*: SVM models map training examples to points in space in order to maximise the width of the gap between two classes. The algorithm uses support vectors to calculate the maximum margin between classes and find a linear correlation between data. For non-linear separation, it is possible to use the *kernel trick*, searching for the boundary in a higher dimension space [58].

- *Artificial Neural Network:* ANN models can be considered the base method of deep learning and aim to mimic the brain's learning process [73]. Its basic element is an artificial neuron as seen in fig 3.1. An artificial neuron is an entity composed of the inputs x and associated weights w, the outputs y and a neuron σ. While σ depends of the pair (x,w), an activation function ϕ is then pondering the output y. Equations 3.2 and 3.3 govern the artificial neuron behaviour.

$$\sigma = \sum_{i=1}^{X_n} w_i x_i \tag{3.2}$$

$$Y = \phi(\sigma) \tag{3.3}$$

  From there, an artificial neural network is a construction of multiple artificial neurons connected in multiple layers. ANN are composed of an input layer, hidden layers and an output layer. Two distinct phases are used during training. First, the *propagation* of the input through the hidden layers gives the prediction available in the output layers. Then these predictions are compared to the real values through a *cost function*. Finally, the *backpropagation* can begin: the error is injected back into the hidden layers, thus modifying the weights w. During the test phase, only the propagation phase is used with the final weights of the training phase.

3.3.2 Unsupervised learning

In anomaly detection problems, the observations labels are not necessary available. In the case of unsupervised learning, the training set is only composed of unlabelled data. The algorithm groups multiple samples into groups sharing similarities called *clusters*. For example, in the general case, let us suppose an unsupervised algorithm is shown many pictures representing a cat, but without labels. In that case, when presented with a picture of a cat, the algorithm cannot tell the subject of the picture, but it might distinguish the content of the picture as similar to the ones offered during training. This can be defined as *clustering*.

Moreover, in anomaly detection, it is assumed that the vast majority of the training observations are normal. In that case, it is possible to predict isolated observations or clusters as anomalous, as shown in figure 3.3b.
Some examples of clustering algorithms are described below:

- *K-means clustering:* This vector quantisation method aims to partition N observations into K clusters [74, 75]. Each point is assigned to the cluster with the nearest mean. K must be chosen by the user. Some methods exist to help decide about the K parameter. Among them, the *elbow method* is the most popular. It consists in calculating the Within Cluster Sum of Squares (WCSS) (see equation (3.4)) for a multitude of K values and then finding the *elbow* on the curve. This break is the optimal number of clusters K

$$WCSS = \sum_{j}^{k} \sum_{x_i \in cluster_j} distance(x_i, C_j)^2 \tag{3.4}$$

  where C_j is the $cluster_j$ centroids and x_i is an observation in cluster $cluster_j$.

- *Hierarchical Clustering:* This method aims to build a hierarchy of clusters [76]. There are generally two strategies. The first one is called *agglomerative*. The algorithm starts with one cluster for each observation and merges them until all data form one unique cluster. The second strategy is called *divisive*. Here, it starts with one single cluster, and splits it recursively as one moves down the hierarchy. As in K-means clustering, the user must specify the number of desired clusters. The dendrogram method is most commonly used to evaluate the adequate number of clusters.

- *Density-Based Spatial Clustering of Applications with Noise (DBSCAN):* DBSCAN is a density-based clustering non-parametric algorithm [77]. It aims to group data points that are closely packed together. It also marks as outliers points that lie in low-density regions. The main parameter to set is the radius of a neighbourhood ε. Note that this algorithm does not need to specify the number of desired clusters.

- *Dynamic clustering for tracking evolving environments (DyClee):* DyClee is a two-stages distance-based and density-based clustering algorithm [78]. Data samples are fed as input to the distance-based clustering stage in an incremental, online fashion, and they are then clustered to form micro clusters. The density-based algorithm analyses the micro-clusters to provide the final clusters. Thanks to a forgetting process, clusters may emerge, drift, merge, split or disappear, hence following the environment's evolution. Like DBSCAN, DyClee does not require the number of desired clusters. Instead, the main parameter to be set is the size of the micro clusters.

3.3.3 Semi-supervised learning and one-class classification (OCC)

Semi-supervised learning is an in-between of supervised and unsupervised learning. It can be complex to define the limit between semi-supervised and unsupervised anomaly detection. Indeed, different definitions can be found in the literature [44, 79]. In this thesis, semi-supervised learning assumes a partial labelling of the training set, or only a partial representation of the classes.

A first problem is multi-class detection that focuses on one class only. Usually, it

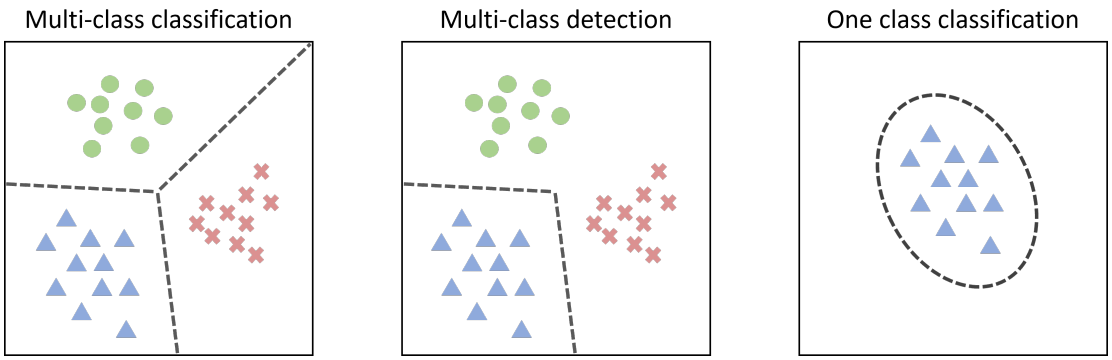


Figure 3.4: Classification overview

is called the positive or target class, in contrast to all other classes called negative or outlier classes [36]. The training is performed with samples from both classes, aiming to separate the positive class from the negative class.

The most common problem is when only observations of one class is available during training. This particular case displayed in figure 3.4 is called *one-class classification (OCC)*. It can be seen as a special case of classification. As shown in figure 3.3c, only the positive class is represented during training in one-class classification. Therefore, the algorithm is trying to find a fitting boundary that models the positive class to leave outside other objects during the inference phase [80].
One-class classification algorithms represent an important asset for anomaly detection. Indeed, it can often prove tedious to gather numerous and representative examples of faulty behaviours. Thus, other classification methods can be nearly impossible to implement due to the lack of positive class. By restraining the need to only normal behaviour during training, OCC is a powerful alternative.

Some examples of well-known one-class classification algorithms are described below:

- *Elliptic Envelope (EE):* EE algorithm aims to encapsulate the training data into an elliptical shape. Then, every data point that falls outside the shape is considered as an anomaly. This method works best with data sets that are Gaussian distributed.

- *Isolation Forest (IF):* IF is an algorithm primary used for outlier detection. Indeed, it is especially efficient for unbalanced data sets by focusing on out-of-distribution samples. An *unbalanced data set* describes the data set for which the number of observations for each class is not equally distributed. This algorithm recursively splits the samples by randomly selecting an attribute and then randomly selecting a split value for this attribute, between its minimum and maximum [81].

- *Local Outlier Factor (LOF):* As the name implies, LOF is an outlier detection method that works by giving a score to each sample of a data set [82]. LOF based its prediction on the density of the neighbourhood. The calculation of the score is a multi-phase algorithm that works as follows. First, the distance between an

observation A and its k^{th} nearest neighbour is calculated. $N_k(A)$ represents the set of k nearest neighbours. Then, a reachability distance is calculated between A and another observation B following equation (3.5):

$$RD_k(A,B) = max(K_{distance(B)}, distance(A,B)) \tag{3.5}$$

where $K_{distance(B)}$ is the distance between B and its k^{th} neighbour. From there, the k-nearest neighbours of each observation have to be found. Then, it is possible to calculate the local reachability density of A using equation 3.6. $LRD_k(A)$ is the inverse of the average reachability distance of sample A from its neighbours.

$$LRD_k(A) = \frac{1}{\frac{\sum_{B \in N_k(A)} RD_k(A,B)}{|N_k(A)|}} \tag{3.6}$$

Finally, it is possible to compare the density of A with its neighbours by calculating the local outlier using equation (3.7):

$$LOF_k(A) = \frac{\sum_{B \in N_k(A)} LRD_k(B)}{|N_k(A)| * LR_k(A)} \tag{3.7}$$

Based on the $LOF_k(A)$ value, the observation A is assigned as outlier or not. Usually, if $LOF > 1$, it is considered as an outlier.

- *One-Class Support Vector Machines:* OCSVM is a special case of SVM [36] [37]. Unlike SVM, which tries to find a hyperplane separating two classes, OCSVM considers a hypersphere encompassing all positive instances. In this case, the margins references outside of the hypersphere. Thus, OCSVM aims to create the smallest hypersphere possible.

- *Auto-Encoders:* AE are a particular case of artificial neural networks where the input layer and the output layer are identical [83]. This architecture gives the ability to the neural network of coding itself. Two phases can be distinguished in AE. The first phase is called encoding. It is the action of transforming the input into the hidden layer space. From there, it is possible to continue the propagation to the output layer. This phase is called decoding. In one-class auto-encoders, the model is is trained to reconstruct a single class. Then a reconstruction error τ is considered during the inference to estimates how well the model is able to reconstruct the input [84]. This error is used as a score to define a new observation into the positive or negative class.

3.4 Distance metric used in anomaly detection

In machine learning and anomaly detection, many algorithms rely heavily on a *distance* metric in order to analyse the similarities between each observation of the input space [85]. Therefore, choosing the distance function that will be computed by the algorithm is a crucial aspect in resolving a machine learning task. Defining this function has been studied for years, and many metrics have been developed to evaluate the distance

between two observations. This section defines the most commonly used metrics for anomaly detection tasks.
For each metric, we assume two points $X = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ and $Y = (y_1, y_2, ..., y_n) \in \mathbb{R}^n$.

- ***Manhattan distance***, also called Taxicab distance, calculates the distance by taking the absolute differences between the points across all dimensions. Even though it is not an intuitive way to measure a distance, it is commonly encountered as an efficient metric in high-dimensional data sets [86]. An example of this metric is displayed in figure 3.5a. Manhattan metric is formulated as in equation 3.8

$$dist_{Manhattan}(X, Y) = \sum_{i=1}^{n} |x_i - y_i| \tag{3.8}$$

- ***Euclidean distance*** calculates the shortest distance between two points. It is the most intuitive and used metric. However, euclidean distance becomes less effective for machine learning applications as the dimensionality of the feature space increases. It is said that euclidean distance works best for three or lower dimensions [87]. An example of this metric is displayed in figure 3.5a. Euclidean distance is formulated in equation 3.9

$$dist_{Euclidean}(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3.9}$$

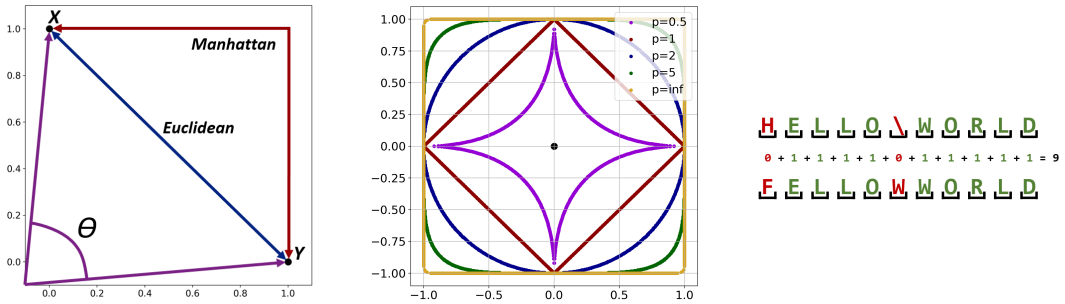
- ***Minkowski distance*** calculates the distance between two points in the normed vector space. It can be seen as a generalised form of the Manhattan and Euclidean distances. It resolves around an order p that must be fixed by the user depending on its application. Usually, its value is a strictly positive integer. Studies demonstrated that choosing $p \in (0, 1)$ does not increase the performance of machine learning algorithms [88]. An example of Minkowski metric for different p values is shown in figure 3.5b. The different curves represent the same Minkowski value from the centre. Minkowski metric is defined in equation 3.10

$$dist_{Minkowski}(X, Y) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{\frac{1}{p}} \tag{3.10}$$

  with $P \in \mathbb{Z}$.
  Note that $p = 1$ refers to Manhattan distance and $p = 2$ refers to Euclidean distance.

- ***Cosine similarity*** calculates the cosine of the angle between two vectors. It is often used in place of euclidean distance for high dimensional data sets. However, it is worth noting that the vectors' magnitude is not taken into account. An example of this metric is displayed in figure 3.5a Cosine similarity is defined in equation 3.11

$$dist_{Cosine}(X, Y) = cos(\theta) = \frac{A.B}{||A||\ ||B||} = \frac{\sum_{i=1}^{n} x_i.y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}} \tag{3.11}$$



(a) Manhattan, Euclidean and Cosine metrics

(b) Minkowski metric

(c) Hamming metric

Figure 3.5: Distance metrics

- ***Hamming distance*** calculates the number of different elements in two vectors. The most common use case is string comparison, by analysing how many characters differ. Note that it is best used when the vectors are of the same length. The hamming distance is calculated using the sum of identical elements between two vectors. An example of this metric is displayed in figure 3.5c.

3.5 Conclusion

In this chapter, A brief history of AI has been presented, then focusing the anomaly detection field . It has been highlighted that three types of anomalies can be encountered. It also has been demonstrated that depending on the severity of the damage, a single event effect can be considered as a point or a collective anomaly. Moreover, some anomaly detection methods have been presented. Divided into three categories, these methods are valid candidates to improve single event effects detection. The focus is now to setup an experimental platform to test anomaly detection methods and evaluate their performance compared to already available detection methods for single event effects.

The first part of this manuscript focused on establishing the theoretical prerequisites for this thesis work. The objective developed in the second part, is to setup an experimental platform to test anomaly detection methods and to evaluate their performance compared to already available detection methods for single event effects.

Part II

Experimental platform

Chapter 4
Experimental circuit design

Contents

To study the effects of radiation on electronic components, one must first select the devices that will be examined. Indeed, the impact of radiation can vary greatly depending on the type of component used [89]. In consequence, performing a study that encloses all types of SEEs might reveal to be a stall. Plus, some components exist in a hardened version specifically designed for space applications [90], so a choice must be made whether to choose the COTS or the hardened version of a component.

The first step of this research was to choose a component sensitive enough to radiation to carry out research on new detection methods. Microcontrollers are complex integrated components. Therefore, it is complicated to carry out a hardened version of a microcontroller, justifying the need for alternative detection methods.
The reference chosen for our study was the ATMEL SAM3X8E. Its normal behaviour must be analysed and documented before any experience in order to compare its evolution in a radiative environment. In consequence, an experimental circuit setup is required.

In this chapter, the goal is to describe the microcontroller chosen to study single event effects. In addition, a specific electronic board designed to go through radiation testing is described.

First, in section 4.1, the component chosen for this study is described. Its main characteristics as well as its behaviour are studied to help the characterisation of single event effects. Then, in section 4.2, the process of creating an electronic board specifically designed for single event effects testing is detailed.

4.1 ATMEL SAM3X8E microcontroller

The ATMEL SAM3X8E microcontroller is chosen as the core component of this study. Therefore, most tests and analyses will be performed on this specific component. It was first considered because the CNES team coordinating this project was already familiar with working with this component. Additional reasons explained in section 4.1.1 validated this choice. Thus, extensive studies on its normal behaviour have been performed in order to have references and prepare radiation experimentations.

4.1.1 Choice reasons

A vast choice of components is available when talking about space equipment. Photovoltaic panels, transistor as switching MOSFET, memory, camera equip satellites on space missions and are all valid options when defining a space case study. Nevertheless, a component able to execute instructions and store informations is needed if it is used to work with a software detection method. This criterion led us to choose a microcontroller as our tested component. The ATMEL SAM3X is finally chosen. Three reasons led to choosing the ATMEL SAM3X over other microcontrollers.

Firstly, this component is currently used in actual space applications. For this particular case, the ANGELS project is taken as an example. ANGELS (Argos Neo on Generic Economical and Light Satellite) corresponds to a first genereation of nano-satellite jointly developed by the CNES and Hemeria. It was successfully launched by a Soyuz launcher on December 18, 2019, from the Guiana Space Center. To control and ensure its functions, the component primary chosen is a SAM3X microcontroller with the 144 pins package. The satellite follows the CubeSat 12U requirements, weighing 20kg. ANGELS satellites are designed to carry Argos-Neo, a whole new generation of instruments set to gather environmental data based on the Argos system.

Secondly, the ATMEL SAM3X8E version is a central component for the widely distributed Arduino DUE development board. Arduino is a well-known company that designs and manufactures single-board microcontrollers kits. Choosing an Arduino DUE board removes the need to design a whole new testing board for preliminary testing, and gives access to a quick setup to test the microcontroller specifications. Furthermore, the Arduino DUE board provides access to numerous IOs allowing communication with the SAM3X8E microcontroller. It also gives the possibility to program the microcontroller using USB communication.

Thirdly, an important criterion in our case study is that the component has to be sensitive to radiation to get failure examples in the study. That is why our choice was to take the COTS version of the SAM3X via the SAM3X8E. Nevertheless, studies show that even the rad tolerant version, the SAMA3X8ERT, is sensitive to single event effects [90]. Indeed, microcontrollers are complex, designing an efficient protection on such components is highly complicated. Therefore, it emphasises the complexity of device protection and the need for new methods to protect space components.

In conclusion, this component is chosen for its use in real space applications, its availability on already existing development boards, and its sensitivity to single event effects.

4.1.2 SAM3X8E specification

The ATMEL SAM3X8E is made by Microchip. The package is an LQFP-144 with 20*20mm² dimensions. It is based on the ARM Cortex-M3 processor, optimised for low-cost and energy-efficient integrated circuits. It possesses 512Kb of flash memory used to store the program. It operates on a 96Kb SRAM at 84MHz. Finally, it features 103 I/O lines. Fore more information, the internal diagram is available in figure B.1 in appendices.
Most of the studies regarding the SAM3X8E were performed using the Aduino DUE development board displayed in figure 4.1.

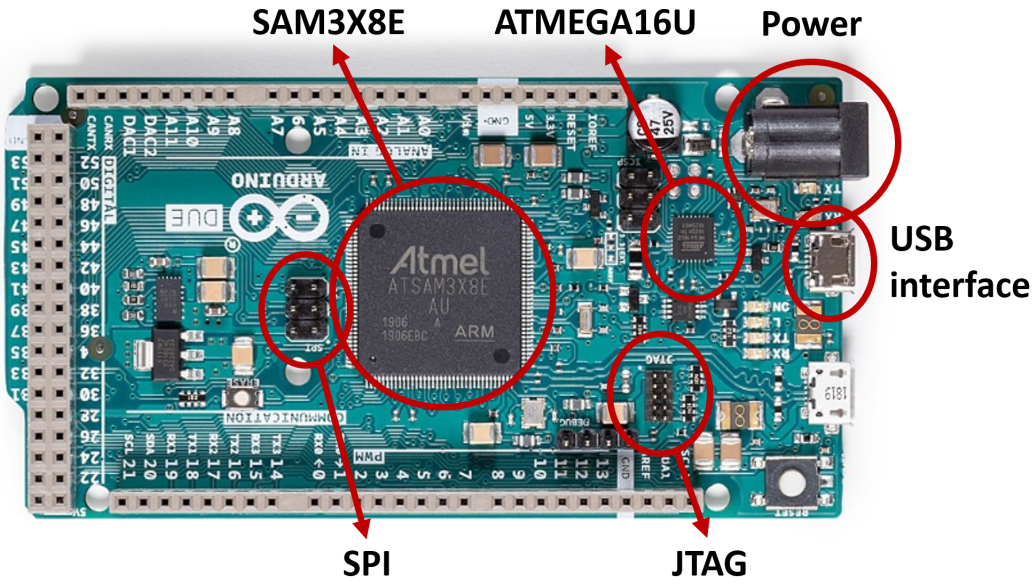


Figure 4.1: Arduino DUE board equipeed with an ATMEL SAM3X8E microcontroller

The Arduino DUE is equipped with a SAM3X8E as its primary component. It also features the ATMEGA16U2, which provides a USB interface between the SAM3X8E and a connected computer. The board is powered using a 7V-12V voltage source. Multiple pins are available and are used to connect with the SAM3X8E I/Os: digital, analogue, PWM, CAN and I2C types are available for a total of 54 I/O pins. For more information on the Arduino DUE, the specifications are available in table B.2 in appendices

It is possible to program the SAM3X8E using the USB port or the JTAG connectors available on the board.

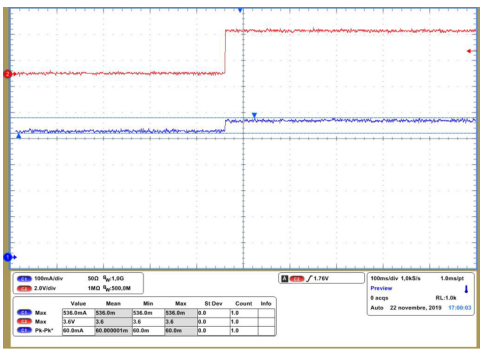
4.1.3 Behaviour study

Understanding the faulty behaviour of a component due to a radiative environment requires a basic understanding of its normal behaviour. The supply current is the main indicator of the component's behaviour when performing radiation testing. Thus, it is the indicator that will be mainly used in this study.
Most ground radiative experimentations are performed using a component in sleep mode. Thus, only a few variations of normal behaviour are available. In that case, detecting non-destructive single event effects is facilitated compared to realistic scenario. In this study, the goal is to provide a complex emulation of the chip behaviour so that it is as realistic as possible. By doing so, some anomalies may not be directly discriminable. This way, it is possible to provide results even for complex cases.

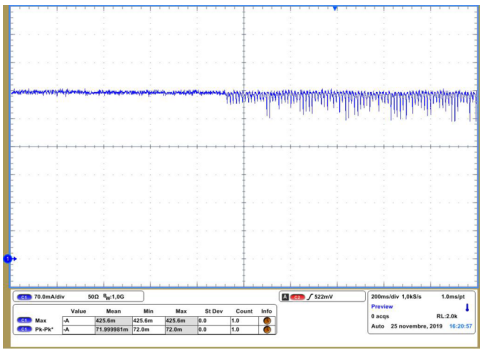
In order to emulate realistic behaviour, different functions are implemented in the microcontroller while monitoring the supply current. Note that the Arduino DUE design was not made to easily access the SAM3X8E's supply current. Thus, only the supply current of the entire board is monitored. However, as the SAM3X8E is the only active component, it is also the only component that significantly impacts the board's supply current during the execution of a program.

During the experiments, two primary consumption profiles were highlighted:

- The ***load consumption profile*** refers to any activities of the microcontroller that results in a modification of the electrical load on the microcontroller. Usually, this results from a state change of an I/O inducing a mean shift of the supply current , as shown in figure 4.2a. The main characteristic of the shift is governed by the resistive value of the attached load.
- The ***software profile*** refers to any modification of a memory cell of the microcontroller. In this case, the supply current is modified depending on the activity of the chip. The mean of the supply current remains the same, whereas the variance is affected, as shown in figure 4.2b. During the experiments, functions such as variable modification, SPI communication or display functions are elements that are considered as a software consumption profile. However, it is difficult to estimate the impact of a specific function on the supply current.



(a) Load profile



(b) Software profile

Figure 4.2: SAM3X supply current profiles

4.2 DIAG-RAD electronic board

During the preliminary phase of this thesis project, the Arduino DUE development board gave crucial information regarding the microcontroller's behaviour. However, harsh radiation testing was planned, and the base functionalities of the Arduino board became limited. It was then decided to develop a specific electronic board in addition to the Arduino DUE.
This board, called the *DIAG-RAD board*, is a dual microcontroller board. It is based on the Université Catholique de Louvain (UCL) cyclotron's frame, as experiments were primarly planned in this facility[1]. The frame's dimensions are displayed in figure B.3 in appendices

4.2.1 Specifications

Before starting the design of the DIAG-RAD board, the testing requirements are listed. It is done to enquire that this board can cover all the needs of radiation testing. The specifications of the DIAG-RAD board are given table 4.1. The functions referred to in the table are described subsequently.

n°	Requirements	Functions
A	- Test of a SAM3X8E - Availability of both front and back sides of the chip	①, ②, ③, ⑥
B	- Compatibility of the board with a testing frame	④
C	- Emulation of various load consumption - Current pulse of 1,2,3,5 or 10 mA	⑧
D	- Emulation of various software consumption	②, ⑧
E	- Protection of the device under test - Protection controlled by a dedicated input - Fault signal must be available as an output	⑤
F	- Monitoring of the device under test - Signal must be numerical output compatible with computer's communication	②, ⑤, ⑦
G	- Availability of Arduino DUE functions to facilitate programming	⑥, ⑦
H	- Emulation of high current events on the supply current - The high current events must be independent of the device under test	⑨

Table 4.1: DIAG-RAD electronic board specifications

[1]However, the reader must not put some expectations into this, as a world sanitary crisis resulting in the closure of the facility are substantial obstacles, that shattered all hopes of the author to see such state-of-the-art equipment.

4.2.2 Characteristics

From the specifications, an electronic board is designed. The characteristics featured on the card are stated next.

4.2.2.1 Mother/daughter boards ①

The DIAG-RAD board is primarily made for experimental tests on microcontrollers. It means that the device under test will be put under extreme stress, and failure of the component is a possibility. However, replacing the whole board for each experiment would be dubious and time-consuming. Therefore, the DIAG-RAD board is composed of two distinct electronic boards: the main test-bench constituted with a *motherboard* equipped with all the functionalities needed for experimental testing, and a *daughterboard*, which includes only the device under test with few passive components. Thus, if a failure occurs on this daughterboard, it would be easily replaced by another daughterboard without damaging other functions. The motherboard and daughterboard are shown in figure 4.3.

Mother/daughter boards respond to requirement A.

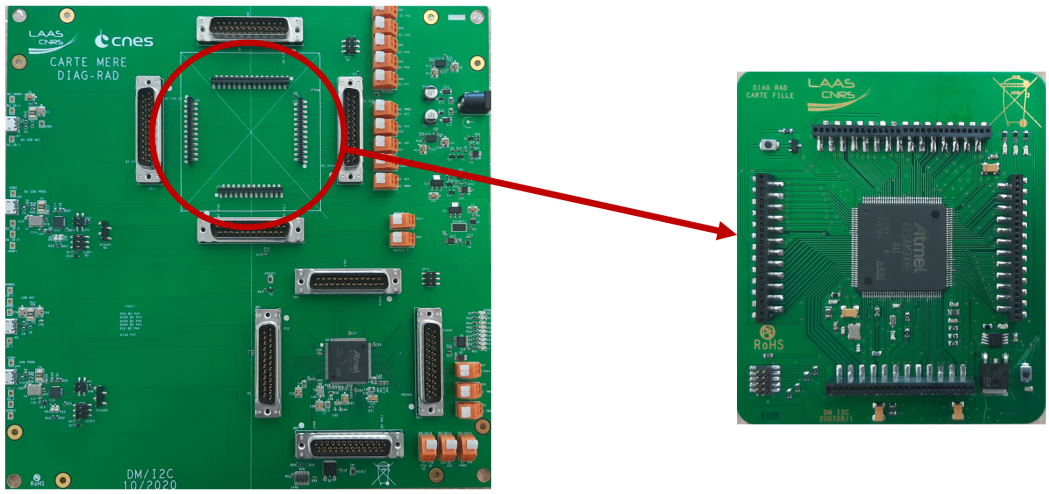


Figure 4.3: DIAG-RAD board (left: mother board; right: daugterboard)

4.2.2.2 Dual microcontroller setup ②

The DIAG-RAD board is equipped with two distinct SAM3X8E microcontrollers. One is mounted on the motherboard, while the other is mounted on the daughterboard. However, each one has a specific role.
The daughterboard's chip is targeted to be the device under test that is going to be monitored and put under stress, whereas the motherboard's microcontroller is used to provide monitoring and supervise tests.

It is possible to communicate between both SAM3X8E microcontrollers as their respective CAN ports are connected. It can be used to gather data on the device under test or to emulate data transfer.

The dual microcontroller specification responds to requirements A, D, F.

4.2.2.3 Flipped daughterboard ③

The specificity of this architecture is that it is possible to plug the daughterboard on both sides of the motherboard. Indeed, male and female board-to-board connectors are present on both sides of each board. Moreover, a hole is drilled on the PCB located at the back of the SAM3X8E microcontroller. Doing so makes it possible to access the back of the chip during a test.

The flipped daughterboard specification responds to the requirement A.

4.2.2.4 Dimensions ④

As specified previously, the DIAG-RAD board was initially designed to fit the frame of the UCL's cyclotron, respecting specifications, connectors and dimensions. Even though, for pandemic reasons, alternative solutions had to be found for radiation tests, it was decided that the overall board dimensions would remain unchanged, as the UCL frame uses standard testing specifications. Therefore, the motherboard is 240*240mm². It features four drilled holes of 6.50mm diameter and eight drilled holes of 3mm diameter used for stability during testing. The daughterboard is 70*80mm².

The dimension specification responds to requirement B.

4.2.2.5 Anti-latch-up system ⑤

Even though the mother/daughter board specificity allows for a quick replacement of a defective chip, it is still preferable to avoid critical failures as much as possible. For this purpose, a protection circuit is designed to be able to power cycle the daughterboard in case of a high current peak. This protection is called an *anti-latch-up system*. The schematic of the protection is available in figure 4.4 .

The circuit is composed of five components. First, a shunt resistor R_s is set up as a current sensor between the power line and the device under test to get the supply current value. Next, a combination of amplifier and comparator is used to compare the supply current with a threshold value. The chosen component to perform these actions is the INA301. Then, the signal is transferred into one port of a NOR gate alongside a software detection input. This input is used to allow an external signal set up by the user to activate the protection. By doing so, it is possible to set up a detection algorithm running alongside the board that can interact with directly the protection. The component used for the NOR gate is the 74LVC1G08GW. The NOR gate signal is

sent into a switch that controls the passage of the current into the daughterboard. The TPS22919 is used for the switch function function.
To summarise, the protection is activated either when the supply current is higher than the set threshold, or when an external signal is sent into the NOR gate.

Moreover, it is possible to monitor the protection's status using the outputs of the INA301 component. Both the supply current value and the logic state of the protection can be retrieved this way.

The anti-latch-up system specification responds to requirements E and F.

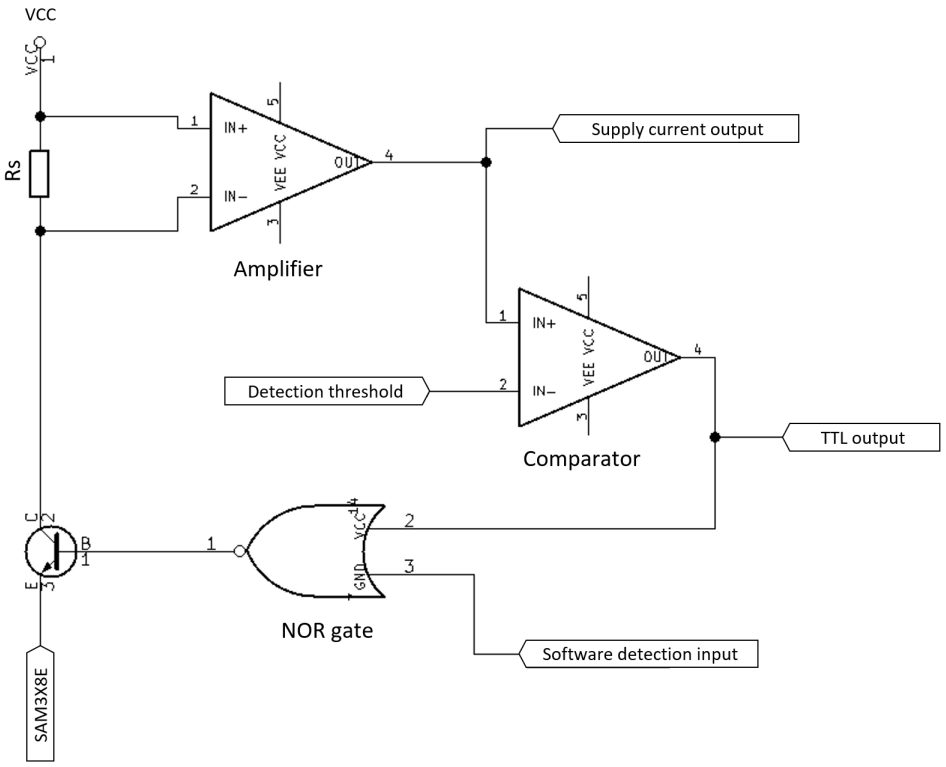


Figure 4.4: Anti latch-up system logic schematic

4.2.2.6 Power supply ⑥

Both boards can be powered simultaneously using a jack connector. The required voltage Vcc to ensure all functions is between 12 to 18V. It is also possible to power supply each board (mother and daughter boards) individually using the corresponding USBs. In this case, the required voltage is standardised to the 5V of a USB port.

The power supply specification responds to requirements A and G.

4.2.2.7 External communication ⑦

For each board, communication with an external computer can be performed in two ways. First, it is possible to program the two SAM3X8E microcontrollers using USB communication. An additional component, the ATMEGA16U2 is used to ensure USB communication between the computer and the chip. It runs at 16MHz and is alimented by 3.3V. From there, it is possible to upload programs written for Arduino boards.

Second, JTAG connectors can be used for each SAM3X8E microcontroller. It allows to upload C programs directly into each chip without the need for Arduino's interface. In this configuration, the Atmel-ICE debugger by Microchip is used as an interface between a computer and the microcontroller.

The external communication specification responds to requirements F and G.

4.2.2.8 Consumption functions ⑧

Consumption functions are set up to emulate a realistic supply current behaviour. To do so, multiple programmed resistive loads are connected at the I/Os of the device under test. They are placed on the motherboard, and linked to the daughterboard by the board-to-board connectors.

A total of ten load slots are available. Five of them have fixed resistors of 50Ω, 100Ω, 250Ω, 500Ω, $1k\Omega$, while the others are resistor plugs that can take in any resistor value needed by the user.

Moreover, analogue pins are linked between both SAM3X8E. Therefore, depending on how the I/Os are configured, they can send signals to each other, which can be seen as a software load.

The consumption functions specification responds to requirements C and D.

4.2.2.9 Fault emulator ⑨

Four fixed loads, characterised by resistors of 50Ω, 200Ω, 300Ω, 500Ω are connected to the 3V3 supply voltage of the device under test. These loads can be controlled by the SAM3X8E on the motherboard. It is done by the TPL7407LPWR, a 7-way switch that can be turned on and off by the I/Os of the motherboard's SAM3X8E.
This way, it is possible to activate an external load on the supply current that does not depend on the device under test behaviour and could be interpreted as a short-circuit.

The fault emulator specification responds to requirement H.

4.3 Conclusion

In this chapter, the SAM3X8E microcontroller that will be used for radiation testing is described. Its specificities are given, and a study on its normal behaviour is performed. Also, two electronic boards are introduced. The first one is the Arduino DUE board that is easily accessible on the market. The second one is the DIAG-RAD board, which was designed and developed during this thesis to meet the specific requirement of this project.

Now that the components and circuit are detailed, the next step is to create an extensive database of both normal and abnormal observations in order to evaluate the performance of machine learning for single event effects detection.

Chapter 5

Data generation and feature extraction

Contents

A crucial aspect of machine learning is having access to qualitative and quantitative database. This aspect might be often overlooked when starting a project in this field, but it does not exist machine learning without data. Moreover, the data must be representative of the modelled system, and in large quantity to be able to train the algorithm correctly.

At the beginning of this project, neither anomaly nor normal data of space component supply current were at our disposal. Indeed, data gathered during space missions are, most of the time, confidential. In addition, even though the CNES is performing extensive testing on various components in heavy-ion facilities, the results are often ordered by private companies, such as Microchip. Thus, the CNES does not hold property over the data log gathered during experimental missions.
The only available files were reports containing graphs of the current consumed by different organs and embedded cards. Even though it can be proved helpful in understanding the impact of radiation on a specific component, it cannot be used to train a machine learning algorithm. Therefore, it is necessary to setup experiments or simulated systems in order to gather experimental observations of single event effects.

Moreover, supply current alone is not sufficient to be able to characterise efficiently non-destructive single event effects. A significant step in machine learning is to extrapolate

meaningful information on the signal. Thus, a study is necessary to extract from the supply current the most valuable features for the characterisation of single event effects.

This chapter aims to answer these three questions:

- *How to gather observations of single event effects?*
- *What features can be used to characterise single event effects?*
- *What data sets will be used to evaluate anomaly detection algorithms?*

In the first part of this chapter, experiments including various testing methods are reported in section 5.1. Following these experiments, a supply current simulator has been developed to constitute an extensive database, which is reported in section 5.2. These two sections aims to answer the first question. After that, the data sets are formalised as time series data streams, as detailed in section 5.3. Using this formalism, it is possible to extract various features that enhance the characterisation of single event effects. The retained features are described in section 5.4. These two sections aim to answer the second question. Finally, the last question is answered in section 5.5, which proposes a description of the databases used to analyse the performance of machine learning for single event effects.

5.1 Experimental tests

The question of having access to anomalous behaviour caused by the radiative environment arose quickly in this project. Indeed, the best case scenario would have been to have access to onboard monitoring of a real space mission. It would have allowed to work and train models on realistic data, giving trusting results. However, as discussed earlier, no databases were at our disposal due to confidentiality restrictions. The only choice left was to perform experimental tests on our own to emulate single event effects. Thus, three types of radiation testing have been performed to collect single event effects observations.

5.1.1 Experimental setup

In order to perform radiation testing, an experimental framework is designed to recover as much data as possible. Therefore, even though the setup for each experiment slightly differs, most of the equipment remains similar. Also, different testing scenarii are developed in order to gather single event effects observations in various situations.

n°	Qty.	Device	Reference
①	2	PC	DELL Precision 7540 Laser control
②	1	Power supply	Tenma 72-8690A
③	1	Data acquisition	Keithley DAQ6510
④	1	Differential multiplexer	7700
⑤	1	Protection device	MAX17612AEVKIT
⑥	1	Device under test	SAM3X8E
⑦	1	Test board	Arduino DUE
⑧	1	Shunt resistor	1Ω
⑨	1	Amplifier	SR560

Table 5.1: Pieces of equipment used in radiation experiments

5.1.1.1 Equipment

The main equipment used for radiation testing is described in this section and referenced in table 5.1. It can be noticed that the specific configuration of the equipment is heavily dependent on each experiment. Thus only key elements are described in this section.

First, a laptop ① is needed for all experiments to supervise the acquisition devices. In addition, another computer is used during the laser experiment to control the laser parameters that will be described in section 5.1.3. This other computer is referred to as *laser control*.

The power supply used for these experiments, identified as ②, is a Tenma 72-8690A. The maximum voltage output is 32V. During the experiments, the Arduino DUE board is powered at 12V, corresponding to data-sheet recommendations.

Data acquisition is performed by the combination of the Keithley DAQ6510 ③ and the 7700 differential multiplexer ④. This combination gives the possibility to measure DC voltage from 100nV to 1000V, and DC current from 10pA to 3A simultaneously. According to the DAQ6510 datasheet [91, 92], the maximum sampling frequency is 333Hz when recording voltage and current simultaneously, while it is 1MHZ when recording a single channel. It is the reason why in most experiments, the voltage and supply current of the DUT are monitored one at a time. The memory used to store measurements is the DAQ6510 standard buffer. Consequently, an individual run is limited by a maximum buffer size of 7.10^6 points.

To avoid the destruction of the DUT, the MAX17612A evaluation kit circuit protection device ⑤ is used. It works as a threshold protection that cuts the power when the supply current exceeds the threshold. The range of the threshold can be adjusted from 10mA to 250mA, which is adequate knowing that the DUT supply current is around 65mA.

5.1.1.2 Test functions

Guided by the two consumption profiles established for the SAM3X8E microcontroller (see section 4.1.3), a total of five functions are created to emulate a complex supply current behaviour. The functions are listed in table 5.2, and detailed subsequently.

n°	Function name	Consumption type
①	oscilloscope	load + software
②	analogue	load
③	addition	software
④	switchPin	load
⑤	USB communication	software

Table 5.2: Functions used during testing

① oscilloscope This function is designed to display the value of an analogue I/O as an oscilloscope would. An LCD screen must be connected to the board through SPI communication to visualise data.
It acts both on the I/O profile, as the load created by the LCD screen may vary depending on the information given, and as a software profile, due to the communication between the chip and the LCD screen.
First, the screen must be connected to the board using the I2C communication pins. Then, during runtime, the value of the monitored analogue I/O is constantly recorded. Depending on the LCD screen dimensions, its value is converted into a height coordinate.

Let us take an input defined by the couple (y_i, Y_i), with $y_i \in \Re_+$ the monitored value of the input signal and $Y_i = (w_i, h_i)$ its coordinates on a screen of dimension $W * H$.
First let us focus on calculating h_i. To do so, two constants $minInput \in \Re_+$ and $maxInput \in \Re_+$ such that $minInput \leq y_i \leq maxInput$ are defined. From there, it is possible to formulate a height ratio as in equation 5.1. Finally, the coordinates Y_i to display on the LCD screen are calculated using equation (5.2)

$$heightRatio = \frac{H}{-\frac{maxInput}{minInput}} \tag{5.1}$$

$$h_i = -\frac{heightRatio}{minInput} * y_i + heightRatio \tag{5.2}$$

The analogue pins have a 10bits resolution. Therefore, the monitored input values are comprised between 0 and 1023.
After finding, h_i, let us focus on w_i. No calculation are necessary for w_i as it can be associated to the index of the latest observation. Indeed, for each new observation, w_i is incremented. Consequently, each index corresponds to a horizontal pixel on the LCD screen. By doing so, each new plot is going to move forward the right axis of the LCD screen. It is considered that most of the screen is travelled when $w_i = 0.75W$. Then, the increment of w_i is stopped, and new observations are replacing the oldest ones.

② analogue This function gives a random value to an analogue I/O. It can be combined with the oscilloscope function by setting up the analogue I/O as the one monitored in this function. The activation on an analogue I/O in addition to the shift of associated bits modifies the load consumption profile of the chip.

③ addition This function constantly adds two variables and stores the result in a third variable. The value of one of the two variables is incremented each iteration. This function influences the software consumption profile by switching the variable's bits at each call.

④ switchPin This function is designed to switch digital I/Os on and off. Doing so is modifying the load. It is possible to use resistors to control the supply current shift due to load consumption. This function impacts the load consumption profile.

⑤ USB communication This function is designed to send messages to a computer connected by USB with the board. The Serial Monitor function of the Arduino IDE is used to display the text. Usually, this function is used as a watchdog during testing, sending a message at fixed intervals to verify the status of the chip. This function impacts the software consumption profile.

Behaviour scenarii During experimental tests, a combination of these functions are used to emulate a realistic supply current behaviour. A total of three test scenarii are created, each one emphasising a specific profile behaviour described in section 4.1.3.

- **Load profile scenario:** This scenario is focuses on the load consumption profile. It regroups functions ② analogue and ④ switchPin. The function ⑤ USB communication is solely used as a watchdog.

- **Software profile scenario:** This scenario focuses on the software consumption profile. It combines functions ① oscilloscope, ③ addition and ⑤ USB communication. Even though the oscilloscope function can be seen as both load and software consumption, it mainly impacts the software profile, while the impact on the load profile is minimal. Moreover, USB communication is also used to send information on some variables at fixed intervals in addition to its base watchdog function.

- **All profile scenario:** This scenario combines both load and software scenarii.

This concludes the presentation of the experimental setup. All pieces of equipment and programs described here are used for the Californium-252 and laser tests performed during this thesis project.

5.1.2 Californium-252 testing

5.1.2.1 TRAD facility

The test facility is located at TRAD Tests & Radiations in Labège, France (see figure 5.1). The site is equipped with a Californium-252 source, a support plate to place the DUT at irradiation distance of the Cf252 source (see figure 5.3a), a frame to fix the support plate, and an airtight tank that isolates the radiation source from the user (see figure 5.3b). As stated in section 2.4, Californium-252 sources emit a small portion of heavy ions [28]. Therefore, it is possible to characterise single event effects by using Cf252 source. The radiation source LET is 42 $MeV.cm^2.mg^{-1}$. The DUT is at 3cm of the radiation source when positioned on the support plate. After the Cf252 support plate is positioned inside the tank, a vacuum is performed at 2.10^{-2}mbar.

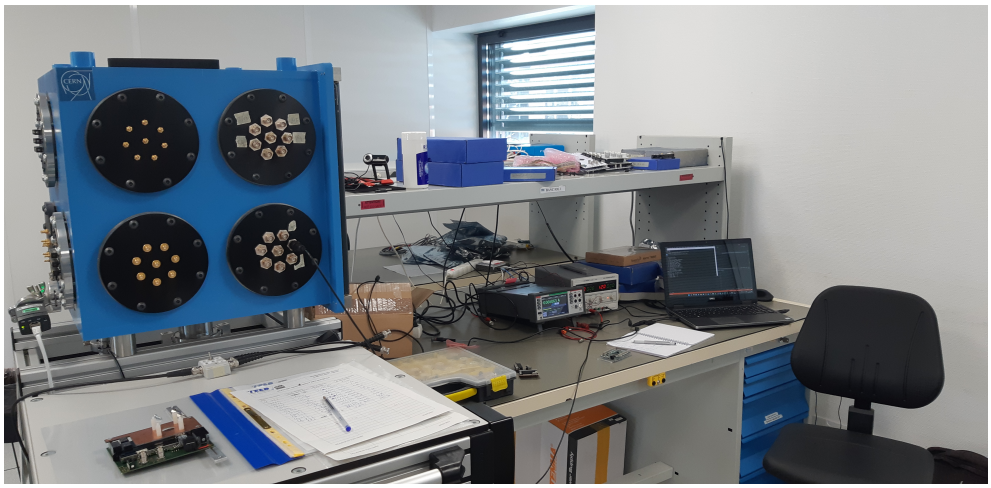


Figure 5.1: TRAD facility

5.1.2.2 Setup

First, the board is positioned at the bottom of the support plate as displayed in figure 5.3a. This plate is equipped with a small motor that is able to cut the DUT from the source. Therefore it is possible to control precisely the exposition time of the SAM3X8E during the whole test. Then, the whole frame is placed into an enclosed tank that isolates the Cf252 source from the surrounding users. The cuve is displayed in figure 5.3b. From there, connections are made accordingly to the schematic shown in fig 5.2 between the board inside the tank and the monitoring equipment. The protection of the MAX17612 device is set to 200mA. Finally, a vacuum is performed in the tank. After that, the test can begin.

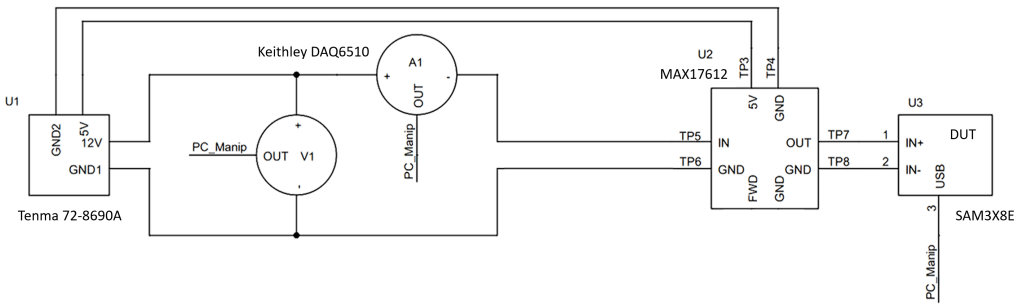
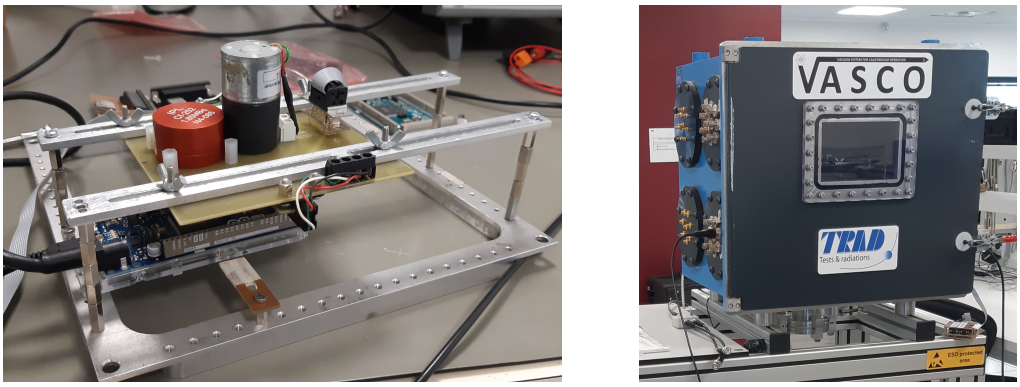


Figure 5.2: Cf252 testing schematic



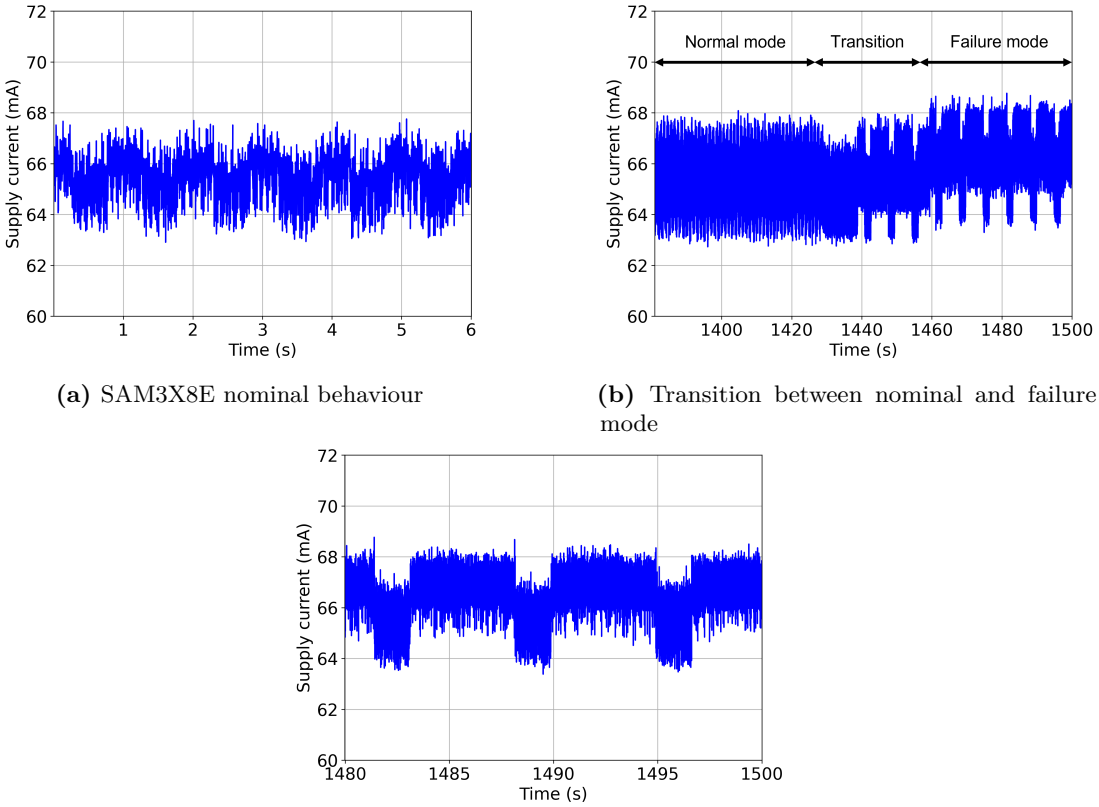
(a) Support plate placed on the frame **(b)** airtight tank

Figure 5.3: Cf252 testing setup

5.1.2.3 Observations

The device under test is powered at 12.0V during the test. Related to this voltage, the DUT nominal supply current is 65mA, with a peak-to-peak value of 4mA. It received irradiation during a total time of 5200s. Around ten high current events that can be seen as single event effects occurred during the experiment. As the maximum supply current value reached in failure mode is 70mA, these faults correspond to a gain of less than 5% from the nominal behaviour. Therefore these faults can be considered non-destructive anomalies and are prone to serve as examples of hidden faults for the machine learning algorithms.

In figure 5.4 is shown an example of a run where a fault occurred. During this run, 1.10^6 points were recorded with a sampling frequency of 666Hz. Figure 5.4a describes the nominal behaviour of the DUT. It represents the beginning of the run and no anomalies have been detected in the supply current. A periodic pattern of 1Hz can be extrapolated from the figure. In addition, when zoomed in, the supply current denotes another periodic pattern of 50Hz.
An anomaly is observed 23 minutes after the beginning of the run. The transition between the nominal and failure mode is captured in figure 5.4c. It can be observed that



(a) SAM3X8E nominal behaviour

(b) Transition between nominal and failure mode

(c) SAM3X8E in failure mode

Figure 5.4: SAM3X8E supply current while expose to the Cf252 radiation source

the transition is not instantaneous. Indeed, a transition phase is seen between 1429s and 1439s, and then between 1439s and 1460s. After that, the failure mode stays up for the rest of the run. Its behaviour is denoted in figure 5.4c. Considerable differences can be pointed out from the nominal behaviour. First, the nominal current value has increased by 1mA. Also, the pattern clearly deviates from the nominal behaviour displayed in figure 5.4a. Finally, the 1Hz and 50Hz periodic patterns have disappeared and been replaced by a new asymmetric pattern lasting around 7 seconds.
It is delicate to give an explanation regarding failure mode behaviour. However, one hypothesis is that when entering failure mode due to an energetic particle, some of the functionalities of the DUT stopped functioning correctly, thus modifying the supply current signature. In addition, with possible local short circuits, it could explain the failure mode's pattern as well as the nominal current increase.

In conclusion, the observations made during this experiment give a good overview of the phenomenon of non-destructive single event effects. However, no destructive anomalies were detected. As stated before, the nominal current only increased by a few milliamperes. Therefore, further experiments are needed to collect more examples of single event effects.

5.1.3 Laser testing

5.1.3.1 The CNES laser facility

Laser testing is a powerful tool when trying to emulate single event effects [30–32]. In this project, The CNES facility in Toulouse is chosen to perform laser testing. As a partner of this thesis project, it is convenient to get access to their equipment. The facility is equipped with a class 1 laser controlled by specific software on a dedicated computer (see figure 5.5).

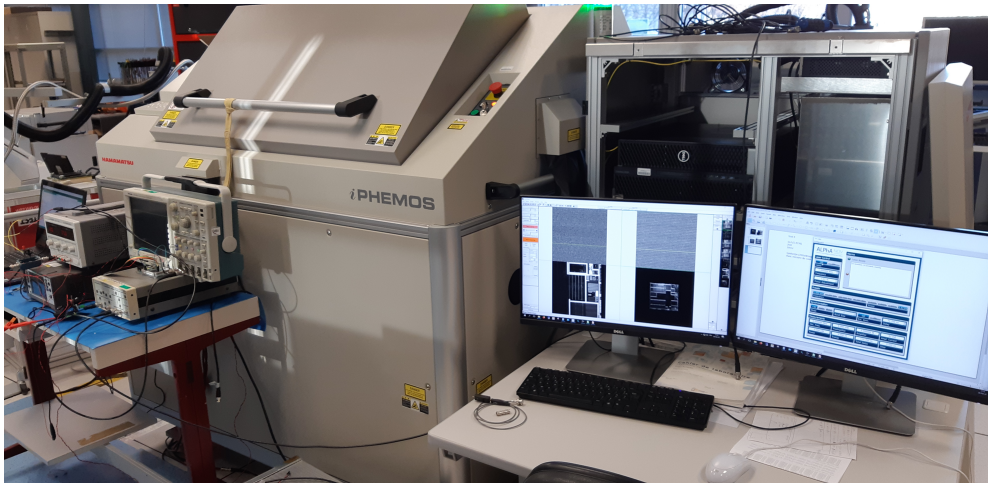


Figure 5.5: CNES laser facility

The characteristics of the laser are available in table 5.3.

Features	Values
Wavelength λ	1064nm
Max pulse frequency	20Mhz
Max power	≈600mW
Optics	1x, 5x, 20x
Scan axis	X, Y

Table 5.3: Laser characteristics

5.1.3.2 Setup

In this experiment, the board is placed inside the laser machine. The schematic of the test is shown in figure 5.6. The setup is similar to the Cf252 experiment, with the exception of the added amplifier as well as an extra Arduino DUE board. This additional equipment is used to monitor the supply current using the SAM3X8E of the Arduino DUE board. The protection device is set to 250mA. Moreover, experiments are performed on the front side of the chip, due to the impossibility of accessing to

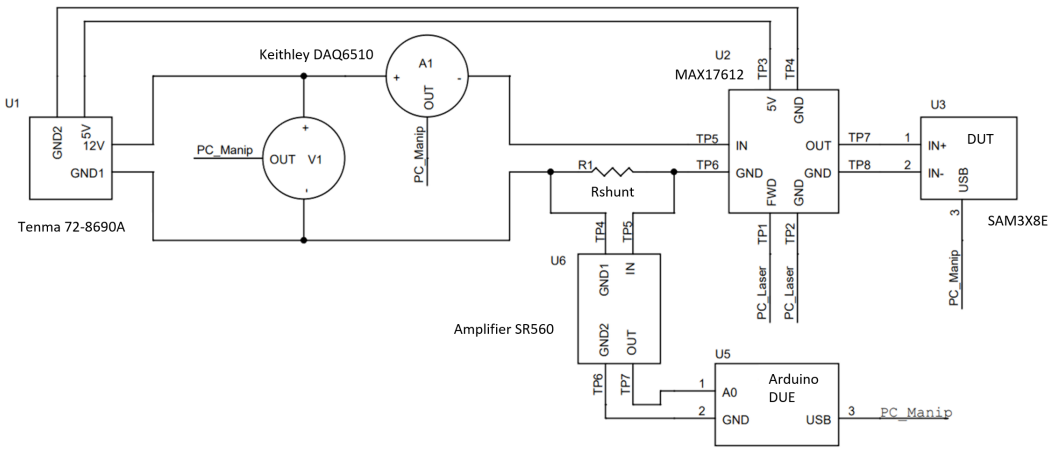
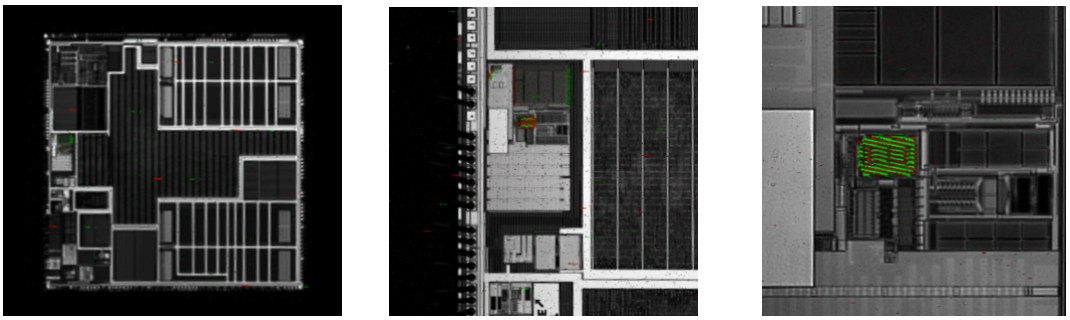


Figure 5.6: Laser testing schematic



(a) Optic 1x **(b)** Optic 5x **(c)** Optic 20x

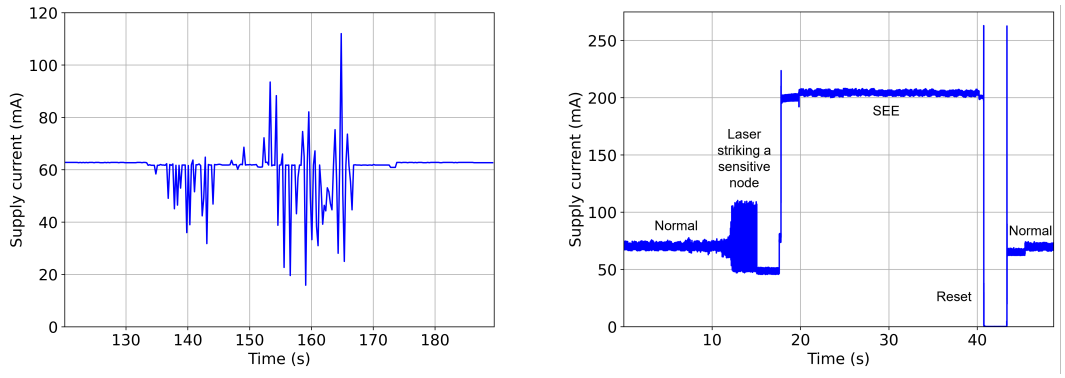
Figure 5.7: Laser picture of the SAM3X8E

the backside of the chip with the tested Arduino DUE board. Using the laser, it is possible to get an image of the chip as it is scanned. It enables the possibility to focus precisely at any sensitive areas discovered during tests. The global picture of the DUT captured by the laser is available figure 5.7a.

5.1.3.3 Observations

A total of three testing campaigns of two days each were performed at the CNES facility, and around ten hours of data were collected. Using the precision provided by laser testing, experimentation's first step was to find sensitive nodes on the DUT. As a result, two sensitive areas were discovered. One of them is highlighted in figure 5.7c.

Two types of faults were discovered during the test campaign. The first one is non-permanent high current events. These faults occurred when the laser is striking exactly a sensitive area of the DUT. In these cases, the supply current returns to normal when the laser stops emitting into the area. Around a hundred of these kinds of faults were



(a) Non-permanent HCE **(b)** Power cycle example

Figure 5.8: Two types of faults discovered during laser testing

recorded. An example of such behaviour is available in figure 5.8.
The second type of fault encountered is permanent high current events. The main difference is that the component stays in failure mode even after the laser stop emitting on the sensitive area. Consequently, it is possible to consider these faults as single event effects. A total of twenty single event effects were recorded during these testing campaigns. After the component enters a permanent failure mode, power cycling is operated to restore the component to its normal behaviour. An example of such a process is displayed in figure 5.8b.

In conclusion, destructive high current events were recorded during the laser test campaigns. Combined with the observations obtained during Cf252 testing, both destructive and non-destructive anomalies examples are available. However, the downside of laser testing is that it can be complicated to assess that the recorded faults are indeed single event effects. Therefore, final experiments are needed to ensure the presence of destructive single event effects to be used with machine learning algorithms.

5.1.4 Heavy ion testing

This experiment has been conducted by the CNES team led by Françoise Bezerra in 2014.

5.1.4.1 The UMCG-PARTREC facility

Heavy ion testing is the primary method used to emulate single event effects. It has been performed in the UMCG-PARTREC facility located in the Netherlands. The PARTREC accelerator facility performs proton and heavy-ion irradiations for radiation hardness testing of electronics or radiobiology. For radiation-hardness testing, the range of energy available for ion testing is 30MeV per Atomic Mass Unit (amu). Also, the flux generated ranges between 10 and 10^5 ions per cm2 for heavy ion testing.

5.1.4.2 Setup - TILU2

An experimental platform has been developed by the CNES in order to perform heavy-ion testing. The result is a piece of equipment acting as a monitoring system as well as a protection device. This device called *Integrated SEL Tester 2nd generation (TILU2)*, can store a recording of the supply current and voltage and pinpoint the location of heavy high current events [93]. This equipment is displayed in figure 5.9.

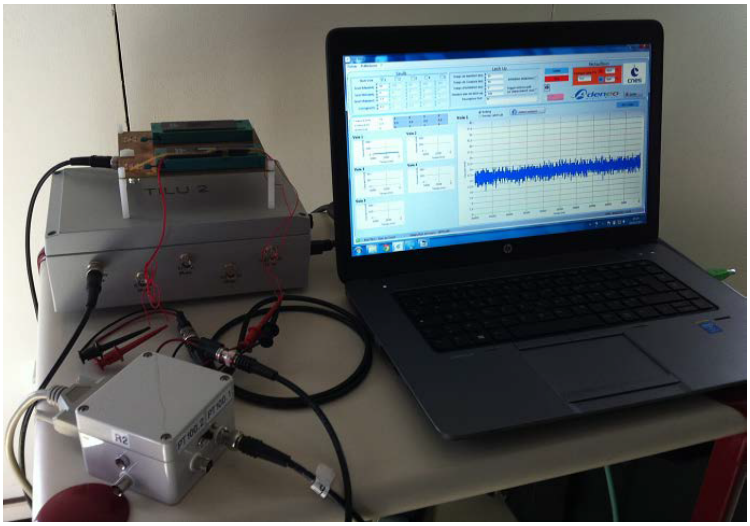


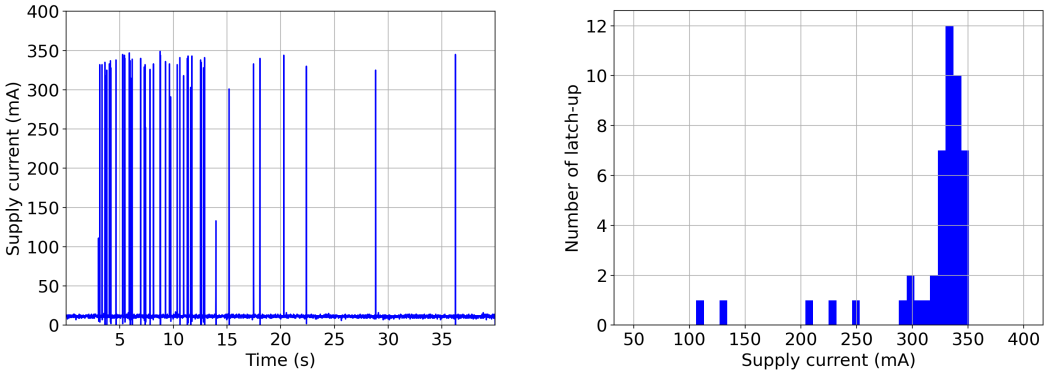
Figure 5.9: TILU2 device (from Bezerra [93])

In addition to the TILU2 equipment, the device under test is a BS62LV4006 CMOS. The sampling time of measures is fixed to 10ms, and the detection threshold is set to 100mA.

5.1.4.3 Observations

A total of 3.10^5 observations have been gathered during 50 minutes. During these runs, the TILU2 threshold has detected 654 high current events. A typical run is displayed in figure 5.10a. The normal value of the DUT is around 15mA. For this specific run, TILU2 triggered 48 anomalies that correspond to single event latch-ups (SEL). When an anomaly is detected, a power reset of the component is done for 30ms. The characteristics of the anomalies detected are given in figure 5.10b. It is possible to see that the vast majority of the single event latch-up resolves around 325mA to 350mA. These values of latch-up could quickly destroy the component if no power reset is performed on short notice. However, in this run, the DUT returned to its nominal behaviour after each power reset of the TILU2, proof that a detection device coupled with a power reset system is efficient in protecting a component against radiation faults.

In conclusion, these experiments give solid observations of single event effects on electronic components. These data can be used to assess the performance of detection algorithms on radiation faults.
However, the number of faults observed is still insufficient to perform statistical analysis regarding the efficiency of a detection system.



(a) A run of TILU2

(b) TILU2 histogram

Figure 5.10: TILU2 run result

5.2 Supply current anomaly simulator

One conclusion drawn after performing experimental testing is that the quantity of observations is insufficient to analyse the performance of machine learning algorithms for the detection of single event effects. More anomaly examples are needed in order to properly exploit the confusion matrix and its derivative indicators described in section 3.3.1. One solution is to use a simulator to provide examples of both normal and abnormal behaviour. Unfortunately, the supply current of a complex component, such as a microcontroller, is influenced by a high number of variables. Passive components values, connected pins or programs booted in the chips are parameters that have to be taken into account to model the supply current. Thus, such a simulator is not available. Most efforts are made to calculate the maximum power consumption of a component, so that it is possible to design a system's power supply. For example, STMicroelectronics proposes its power consumption calculator for STM32L components.

In consequence, a supply current simulator that is able to insert high current events is developed in MATLAB for this project [94]. The data gathered from the experimental testing are used as examples to mimic the behaviour of a real microcontroller.

The advantage of using a simulator is to have complete control over the generated data. It is possible to know precisely where the anomalies are located, improving the quality of the results analysis. Also, it is possible to create numerous scenarii regarding the supply current behaviour of a microcontroller, as well as being able to test on both destructive and non-destructive anomalies.

The simulated data set considered can be defined as $I(t) = \{x(t), y(t)\}$ where $I(t) \in \Re_+$ corresponds to the supply current value at time t, and $y(t) \in \mathbb{B}^n$ is a vector identifying the active functions at a time t. The variable $I(t)$ results from multiple functions and is expressed as in equation 5.3. The total number of active functions corresponds to the size of $y(t)$.

$$\begin{aligned} I(t) = (f_{Nominal}(t, \bar{I}) + \sum_i f_{Load_i}(t, l_i, t_i, d_i) \\ + \sum_j f_{Soft_j}(t, s_j, t_j, d_j) + \sum_k f_{Devi_k}(t, a_k, t_k, d_k) \\ + \sum_l f_{HCE_l}(t, f_l, t_l, r_l)) * \sum_m f_{Reset_m}(t, t_m, d_m) \end{aligned} \tag{5.3}$$

with $i, j, k, l, m \in \mathbb{N}^+$ indexing the multiple occurrences of each function.
The $f_{Nominal}$ function simulates the base current of a microcontroller. It is done by taking the mean current of the component $\bar{I}$ and adding uniformly distributed noise:

$$f_{Nominal}(t, \bar{I}) = \bar{I} + Noise(t) \tag{5.4}$$

The f_{Load_i} function simulates the electrical load created by a component on the microcontroller. It is defined as a rectangular function:

$$f_{Load_i}(t, l_i, t_i, d_i) = \begin{cases} 0 \text{ if } t < t_i \text{ and } t > t_i + d_i \\ \frac{l_i}{2} \text{ if } t = t_i \text{ or } t = t_i + d_i \\ l_i \text{ if } t > t_i + d_i \text{ and } t < t_i + d_i \end{cases} \tag{5.5}$$

with l_i the added load current, t_i the time when the load begins, d_i the duration of the electric load.
The f_{Soft_j} function corresponds to the current modifications induced by the internal processing of the microcontroller (calculations or memory modification for example). When active, this function amplifies the noise already present in the $f_{Nominal}$ function:

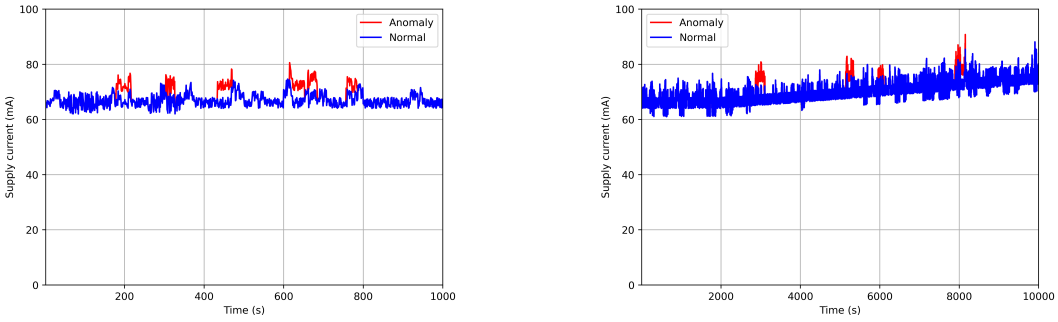
$$f_{Soft_j}(t, s_j, t_j, d_j) = \begin{cases} 0 \text{ if } t < t_j \text{ and } t > t_j + d_j \\ -1^{r(t)} * s_j \text{ if } t \geqslant t_j \text{ and } t \leqslant t_j + d_j \end{cases} \tag{5.6}$$

with s_j the added noise, $r(t)$ a function alternating between 0 and 1, the random function, t_j the time when the function begins, d_j the activation duration of the function.
The f_{Devi_k} function corresponds to a slow deviation of the component's normal behaviour. It can be caused by a change in the component environment such as temperature variation or ageing. A linear function is then applied to the data set:

$$f_{Devi_k}(t, a_k, t_k, d_k) = \begin{cases} 0 \text{ if } t < t_k \\ a_k(t - t_k) \text{ if } t \geqslant t_k \text{ and } t \leqslant t_k + d_k \\ a_k(t_k + d_k) \text{ if } t > t_k + d_k \end{cases} \tag{5.7}$$

with a_k the linear coefficient of the variation, t_k the time when the variation begins, d_k the duration of the function remaining active..
The $Reset_m$ function simulates power cycling:

$$f_{Reset_m}(t, t_m, d_m) = \begin{cases} 1 \text{ if } t < t_m \text{ and } t > t_m + d_m \\ 0 \text{ if } t \geqslant t_m \text{ and } t \leqslant t_m + d_m \end{cases} \tag{5.8}$$



(a) Simulated test set

(b) Simulated deviation set

Figure 5.11: Simulator examples

with t_k the time when the reset begins, d_k the duration of the function remaining active.. Finally, the f_{HCE_l} function simulates persistent high current events. It is similar to f_{Load_i}, except that the modification stays active permanently or until f_{Reset_m} is performed:

$$f_{HCE}(t, f_l, t_l, r_l) = \begin{cases} 0 \text{ if } t < t_l \text{ and } t \geq r_l \\ f_l \text{ if } t \geq t_l \text{and } t < r_l \end{cases} \tag{5.9}$$

with f_l the HCE magnitude, t_l the time when the fault begins, r_l the time of the next reset (the end of the data set if no reset occurs afterwards).

Examples of data sets generated with this simulator are shown in figure 5.11

5.3 Time series data stream

The data sets of the supply current gathered in the previous section can be referenced as a *time series*. A time series X is a data set in which each observation x is indexed by a time t. Time series are defined in equation 5.10

$$X = \{x_t, t \in T\} \tag{5.10}$$

with $T \in \Re_+$ is the index set of X.

As this work is focused on real-time application, not all observations are available at once. Then, a special case of time series called *time series data stream* is used [95]. When all observations are always available in a time series, only previous observations $[x_{t-\Delta t}, x_t]$ are known in a time series data stream. New observations must be treated on the fly, and a continuous update is therefore necessary.

Finally, the definition of a time series data stream is extended to take the labels needed in anomaly detection into account. Thus, the data set used in the following chapter can be defined as in equation 5.11

$$X = \{x_t, y_t, t \in T\} \tag{5.11}$$

with $y_t \in \{-1, 1\}$ the label of the observation x_t. $y_t = -1$ means that x_t is an anomaly, and $y_t = 1$ means that x_t is normal.

5.4 Feature extraction

The characteristics of an individual observation are called features. It is the only information given to a machine learning model about the studied system. Therefore, choosing them wisely is a critical step in data science. No relevant features and the algorithm will be unable to properly model the system, while too many redundant or unnecessary features introduce noise. In both cases, poor feature choices induce-low quality predictions. In this context, the first step in introducing machine learning is to discover which features must be chosen to discriminate between normal behaviour and single event effects.

In the case of single event effects, the monitored data consists of time series of various indicators such as supply current, supply voltage or device temperature. In this study, efforts are focused on the supply current, as it is the most common indicator in single event effects evaluation. However, supply current value alone is insufficient to perform accurate predictions using machine learning models. Additional features have to be created on the time series to leverage as much information as possible. The process of extracting information from a feature space onto a new one is called *feature extraction*. Afterwards, it is possible to analyse the relevance of the extracted features and eliminate the ones that do not give valuable information using a process called *feature selection*.

The data sets gathered through Cf252 and laser testing are used for feature selection. Therefore, the remarks given in the following sections are valid for both experiments. Moreover, it is important to note that this preliminary study is heavily application-dependent. Nonetheless, tests performed afterwards on the data sets provided by CNES on different components proved that it is possible to extend these features to more applications aiming at characterising single event effects.

Statistical analysis, as well as frequency analysis, are put to the test for the characterisation of single event effects [96]. Normal behaviour is compared to failure mode using these features to establish their relevance.

5.4.1 Statistical features

Statistical evaluation is a classic process in time series analysis. It is performed on a collection of observations to uncover trends or patterns in the data set. Thus, finding a method to group observations is required. In the case of a finite time series, one of the possibilities is to perform statistical analysis on the entirety of the observations. However, as stated in section 5.3, not all observations are known in the case of time series data streams.

To address this issue, the *sliding time window* method is used by creating groups of observations. Let us consider a window function h_t, such as the sliding time

window of a time series data stream becomes $H = \{x_t.h_t, t \in T\}$. In this work, h_t is defined as in equation 5.12

$$h_t = \begin{cases} 1 \text{ if } t \in [\tau - \Delta t, \tau] \\ 0 \textit{ otherwise} \end{cases} \tag{5.12}$$

with τ the time of the latest observation x_t and Δt the window's size.

By doing so, statistical analysis is performed by computing the statistical features of the whole observations such as $x_t.h_t \neq 0$. Statistical features are then calculated for each observation x_t.

In this study, many features were calculated, but only a small part gave a good characterisation of single event effects. Therefore, a feature selection is performed using the parallel coordinates [97] as a tool to discern decisive features in the characterisation of single event effects. This method plot all observations to visualise each feature independently. It is then possible to select the most discriminant features by comparing their values for each classes.

Four of them are kept for their relevance: the mean, the variance, the standard error of the mean and the median absolute deviation. Figure 5.12 displays the parallel coordinates of these four indicators for destructive anomalies (figure 5.12c) and non-destructive anomalies (figure 5.12d). Independently of the type of anomaly, it is shown that these indicators enable to discriminate between normal and anomalous observations.

5.4.1.1 Arithmetic mean

The arithmetic mean of a sliding time window H_X is defined in equation (5.13):

$$\bar{H}_X = \frac{1}{n} \sum_{t=\tau-\Delta t}^{\tau} x_t \tag{5.13}$$

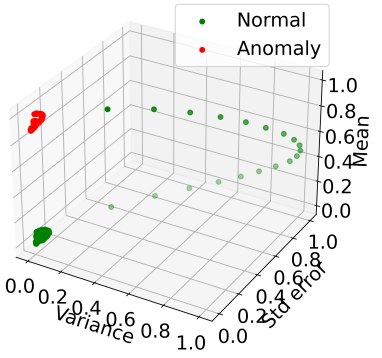
with $n \in \mathbb{N}$ the number of observation x_t *with* $t \in [\tau - \Delta t, \tau]$.
When characterising a persistent anomaly such as a single event effect, a study of the mean value is a relevant indicator. Indeed, an abrupt shift of the signal mean might indicate an anomaly, as displayed in figure 5.13a. However, if the mean shift is small enough, it can be tricky to discern the anomaly from normal behaviour. Thus, other criteria have to be found.

5.4.1.2 Variance

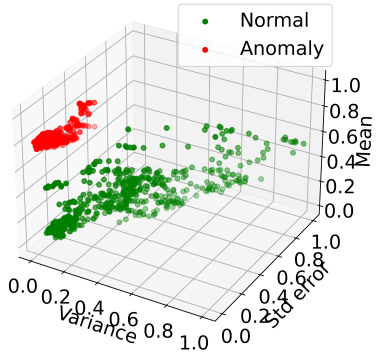
The variance of a sliding time window H_X is defined in equation (5.14):

$$\mathbb{V}(H_X) = \frac{1}{n} \sum_{t=\tau-\Delta t}^{\tau} (x_t - \bar{H}_X)^2 \tag{5.14}$$

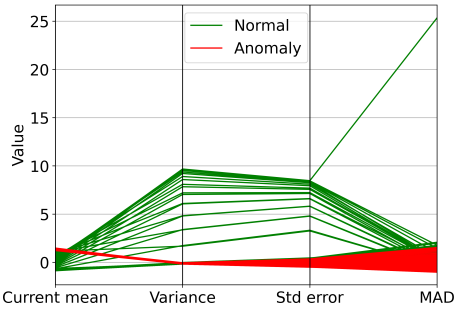
The variance can be used to detect the beginning of a SEE. Indeed, the variance is heavily affected by the mean shift induced by a heavy ion. Therefore, a local increase of the variance can be a symptom of a single event effect, as shown in figure 5.13b.



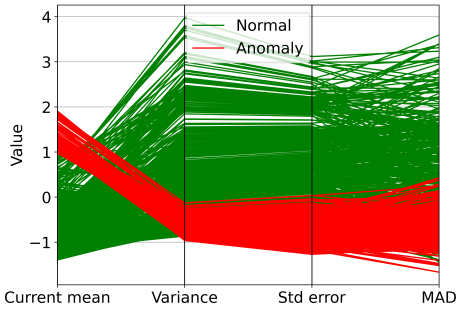
(a) Statistical features for destructive anomalies



(b) Statistical features for non-destructive anomalies



(c) parallel coordinates for destructive anomalies



(d) parallel coordinates for non-destructive anomalies

Figure 5.12: Parallel coordinates

5.4.1.3 Standard error of the mean and median absolute deviation

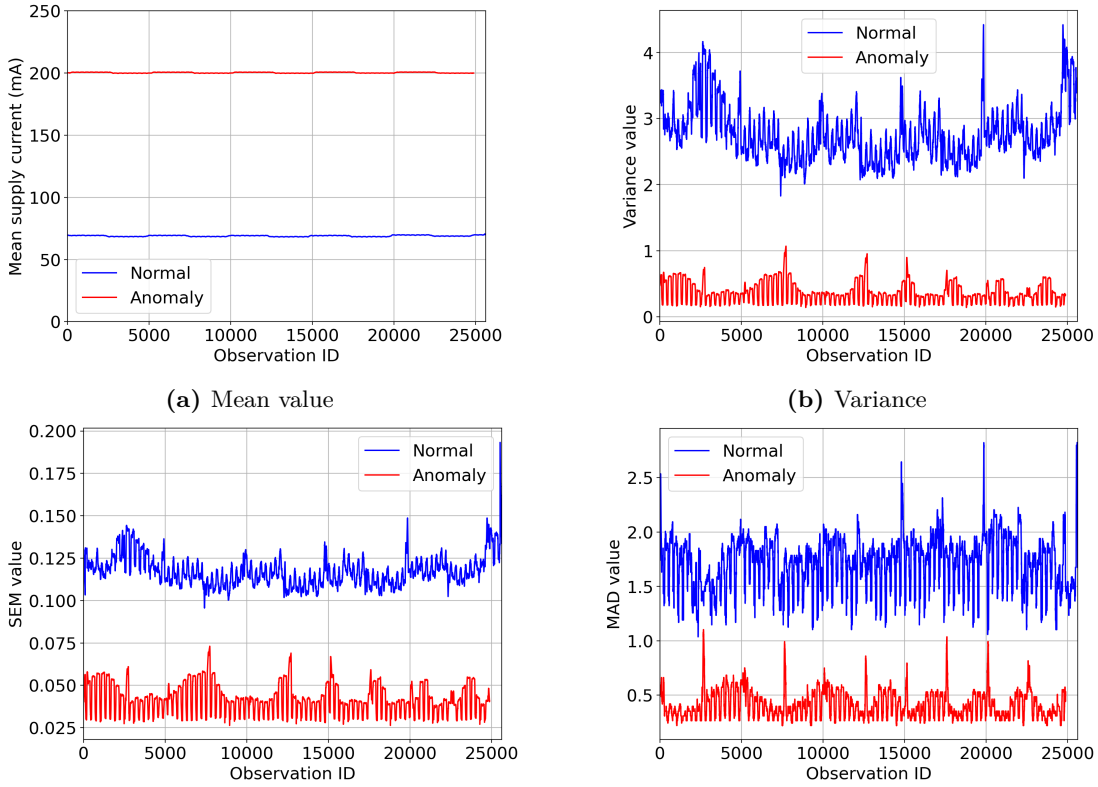
The Standard error of the mean (SEM) of a sliding time window H_X is defined in equation (5.15):

$$\sigma_{H_X} = \frac{\sqrt{\mathbb{V}(H_X)}}{\sqrt{n}} \tag{5.15}$$

The median absolute deviation (MAD) of a sliding time window H_X is defined in equation (5.16):

$$MAD(H_x) = median(|x_t - \tilde{H}_X|) \tag{5.16}$$

with $\tilde{H}_X$ the median of the sliding time window's observations.
These two criteria are chosen due to analysis of the impact of a single event effect on the microcontroller functionalities. Indeed, in failure mode, functions, such as communication with the computer, are disabled deeding on the stroked sensitive node of the microcontroller. Therefore, this loss of activities reverberates through the supply current profile. It is possible to measure this phenomenon by using the standard error of the mean and the median absolute deviation. Then, a decrease in these indicators can characterise a SEE, as displayed in Figures 5.13c and 5.13d.



(a) Mean value

(b) Variance

(c) Standard error of mean

(d) Median absolute deviation

Figure 5.13: Statistical features

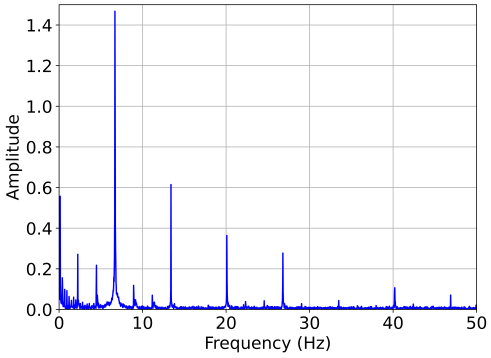
In conclusion, using of statistical analysis could be key in characterising small and hard-to-detect single event effects. By offering the advantage of being easily interpretable, as well as fast to compute, the proposed indicators are fitted to improve the detection of single event effects.

5.4.2 Frequency based features

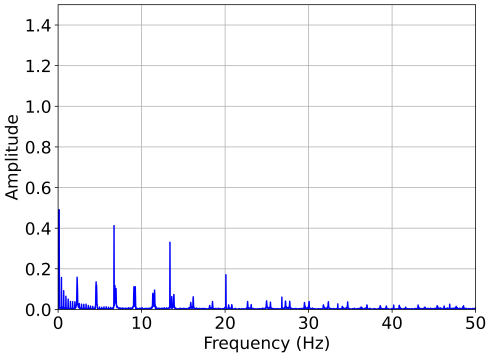
In addition to statistical features, an analysis of the frequency spectrum of single event effects is performed. Using the *discrete Fourier transform (DFT)*, it is possible to obtain the frequency spectrum of finite sequences. In this work, the Fast Fourier Transform (FFT) algorithm is used to calculate the DFT of a data set, as defined in equation (5.17):

$$\mathcal{F}_k(H_X(t)) = \sum_{t=\tau-\Delta t}^{\tau} x_t.e^{\frac{-2i\pi}{n}kt} \; for \; k \in [\tau - \Delta t, \tau] \tag{5.17}$$

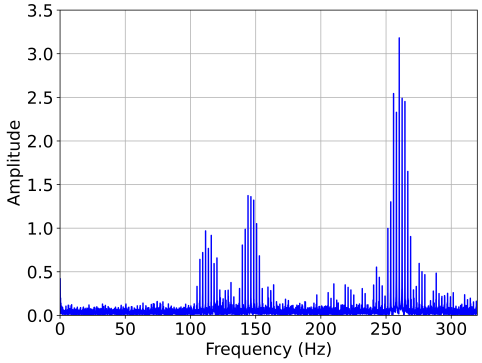
By doing so, it is then possible to compare the frequency spectrum of normal behaviour with anomalous behaviour. However, note that these results only consist of a preliminary study on limited data sets. Indeed, an extensive study by multiplying radiation condition



(a) Frequency spectrum for normal behaviour



(b) Frequency spectrum for faulty behaviour



(c) Frequency spectrum during a laser pass on a sensitive node

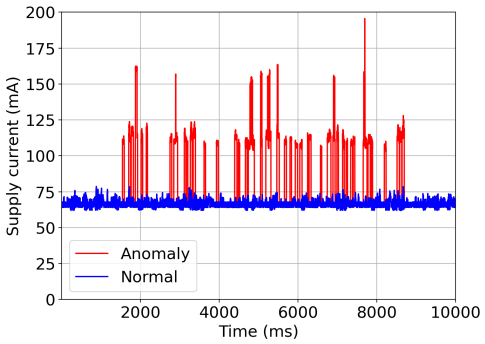
Figure 5.14: Frequency spectrum

tests is needed to define the impact of single event effects on the frequency spectrum more precisely.

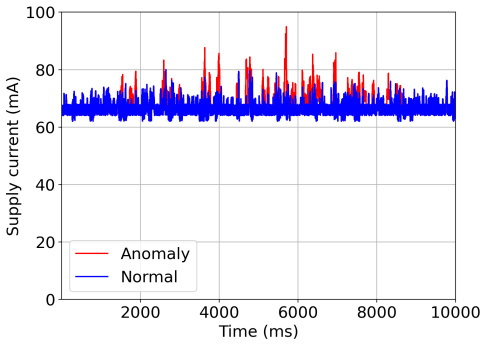
In this study, the frequency value corresponding to 0Hz is removed. Indeed, as the amplitude of this frequency is far greater than any other, it obstructs the readability of the resulting graph.

Figure 5.14a corresponds to the spectrum of the SAM3X8E during its nominal behaviour. The prominent harmonic is located around 6.5Hz, and is echoed at 13Hz, 20Hz and 26.5Hz. Minor peaks are also visible at 2Hz, 4Hz and 40Hz. Aside from these peaks, there is little noise to be seen on the spectrum.
On the other hand, a frequency spectrum created from an active single event effect data set is displayed in figure 5.14b. A significant decrease in the prominent peaks (6.5Hz, 13Hz, 20Hz and 26.5Hz), as well as secondary peaks (2Hz, 4Hz and 40Hz) can be observed compared to the same frequencies of the normal behaviour. Moreover, many small noisy peaks appear when the component is in failure mode.

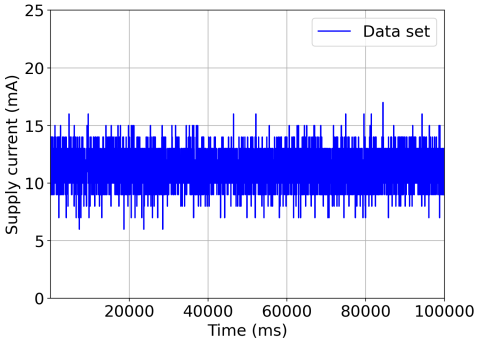
Also, an interesting phenomenon was noted during laser test protocols. A particular spectrum emerges when the laser is pointed at a sensitive node, as displayed in figure



(a) Simulated training set of destructive anomalies



(b) Simulated training set of non-destructive anomalies



(c) Heavy ion training set

Figure 5.15: Training database examples

5.14c. The presence of multiple peaks in the higher range of frequencies was systematic during these experiments, and could be valuable information to characterise the start of a single event effects. This behaviour analysis is possible thanks to the laser capability to unearth specific locations on the microcontroller.

In conclusion, frequency features could be a new lead to improve the detection of single event effects. It has been seen that the frequency spectrum of the nominal behaviour and single event effects significantly differ, and that it would be possible to pinpoint the beginning of an anomaly by analysing the amplitude of high frequency peaks.

5.5 Databases description

Previous sections have presented the means deployed in creating an extensive database grouping examples of both destructive and non-destructive single event effects. These data are to be used to test and analyse the performance of machine learning algorithms for the detection of single event effects. Thus, data sets have been selected for this only purpose. Divided between the training set and test set, this section presents an exhaustive list of all data used in the following chapters.

5.5.1 Training sets

The training set contains the observations used to fit the parameters of the machine learning model. In table 5.4 are referenced the data sets used to train the models. A total of three different training databases are used to fit machine learning models. All data sets used are composed of the supply current of an electronic component as the sole monitored feature. Note that additional features can be computed on the fly when executing a machine learning algorithm.

$Train_{simD}$ and $Train_{simND}$ both stand for data sets created using the simulation described in section 5.2. The former includes observations of destructive anomalies (the supply current in failure mode is around two times higher than the supply current in normal mode, as shown in figure 5.15a), while the latter includes observation of small hidden faults (a few milliamperes higher than normal mode, as shown in figure 5.15b).

$Train_{sim}$ stands for simulated data sets that do not include any fault. Only normal observations of the supply current are available. This database is used to train one-class classification algorithms (see section 3.3.3).

$Train_{HIon}$ stands for the data set created using the data collected in the heavy ion facility, as described in section 5.1.4. Like $Train_{sim}$, this training database does not include any fault observation, as it is used to train one-class classification algorithms. An example of this training set is available in figure 5.15c

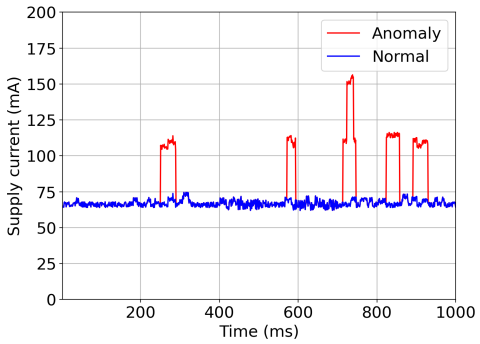
Table 5.4: Training sets overview

	$Train_{simD}$	$Train_{simND}$	$Train_{sim}$	$Train_{HIon}$
Number of set	10	10	10	2
Data per set	1.10^4	1.10^4	1.10^4	$[1, 4.10^3,\ 1, 7.10^4]$
Total data	1.10^5	1.10^5	1.10^5	$5, 4.10^4$
Positive rate	21.03%	21.37%	0.00%	0.00%

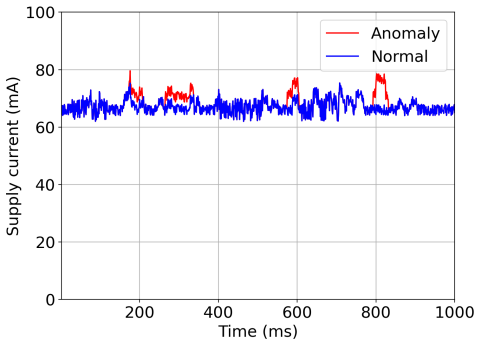
5.5.2 Test sets

The test sets contain the observations that will be used to analyse the performance of a trained model. To do so, a differentiation is made between the true label y_t and the predicted label $\tilde{y}_t$ given by the model. Afterwards, it is possible to create the confusion matrix and resulting indicators, as stated in section 3.3.1.
A total of five databases are used to evaluate the different models studied in chapter 6. Four of them are coming from the simulation software (see section 5.2), while the fifth is from heavy ion testing. Again, the characteristics of all databases are available in table 5.5.

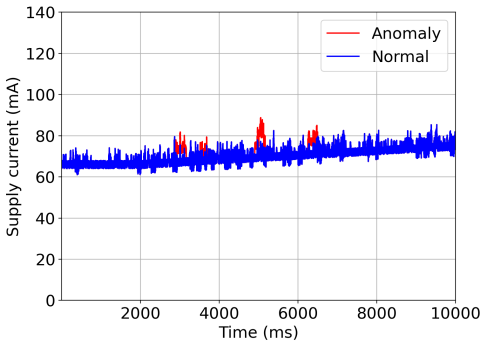
$Test_{simD}$ and $Test_{simND}$ are both simulated databases. Similarly to the train sets



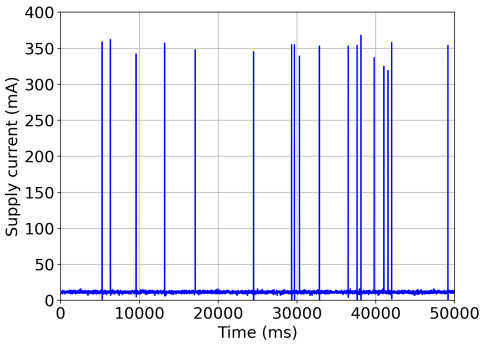
(a) Simulated test set of destructive anomalies



(b) Simulated test set of non-destructive anomalies



(c) Simulated test set of non-destructive anomalies



(d) Heavy ion test set

Figure 5.16: Test database examples

described in section 5.5.1, $Test_{simD}$ contains observations of destructive faults, while $test_{simND}$ contains observations of non-destructive faults. An example of these databases is displayed in figures 5.16a and 5.16b. Regarding $Test_{simDevD}$ and $Test_{simDevND}$, these databases provide similar faulty observations as the two databases described earlier. In addition, a linear deviation is added throughout these data sets by using the deviation function f_{Devi_k} of the supply current anomaly simulator (see section 5.2). An example of a data set of $Test_{simDevND}$ is provided in figure 5.16c. As their nature of simulated data set, the label y_t of an observation x_t is always known. Thus, it is possible to perform a complete analysis of the model's performance.

$Test_{HIon}$ database is coming from the data collected by the CNES during the heavy ion campaign. In order to get the positive rate value, a threshold on the supply current value has been placed to count the number of single event effects recorded. In this campaign, the value was fixed at 50mA, thus only faults that would create a high current event higher than 50mA are counted. In consequence, unlike simulated data, it is not possible to assess the exact number of faults, as only significant single event effects can be numbered.

Table 5.5: test sets overview

	$Test_{simD}$	$Test_{simND}$	$Test_{simDevD}$	$Test_{simDevND}$	$Test_{HIon}$
Number of set	20	20	10	10	8
Data per set	1.10^3	1.10^3	1.10^4	1.10^4	$[4, 0.10^3,\ 6, 5.10^4]$
Total data	2.10^4	2.10^4	1.10^5	1.10^5	$3, 1.10^5$
Positive rate	16.99%	16.39%	9.06%	9.56%	0.21%

5.6 Conclusion

In this chapter, multiple experiments have been conducted in order to establish an extensive database supply current observations. Laser, californium-252 and heavy ion testing are performed to get a wide range of single event effects observations. From destructive anomalies with laser and heavy-ion testing to non-destructive anomalies with californium-252 testing, a solid database of single event effects is at our disposal. Moreover, by using this knowledge, a simulation of the supply current is developed based on the knowledge gathered from the experiments. This simulation gives the possibility to create an infinite number of diverse data sets.

Next, feature extraction is performed on the SAM3X8E's observations to extract additional information regarding single event effects impacts on the supply current. Statistical indicators as well as the frequency spectrum can be considered as valid features for the characterisation of single event effects.

Finally, by regrouping all previous observations, various databases have been created that regroup both normal and abnormal observations in various situations.

This chapter concludes the second part of this manuscript. A solid database regrouping single event effects observations has been created. Using this database makes it possible to evaluate the performance of anomaly detection methods for the specific case of single event effects, what is the aim of the last part of this thesis.

Part III

Machine learning based anomaly detection for electronics hardening

Chapter 6

Machine learning feasibility for space mission reliability

Contents

Single event effects detection is a crucial aspect of a space mission. Most of today's techniques are focused on hardware detection methods that are tedious or expensive to implement. However, software detection methods are yet to be investigated. The ambition of this chapter is to demonstrate that machine learning is efficient for the detection of single event effects. The first action is to perform a proof of concept of these methods. On that account, well-known machine learning algorithms are tested for single event effect detection using the databases described in section 5.5.

As machine learning is not yet used for the single event effect detection problem, it is necessary to set performance thresholds that must meet the algorithms to be accepted. Machine learning is compared to the baseline threshold detection method to establish the requirements.

Referencing section 3.3, it is possible to divide anomaly detection into several categories. The border between each category is the information given by the training set [44]. Is it possible to observe all classes? Are the labels available? From these questions, it is possible to separate the different methods. It is based on these information constraints that the different tests went through.

This chapter aims to answer these three questions:

- *How to judge the performance of a detection algorithm?*
- *Is anomaly detection efficient for single event effects detection?*
- *What are the minimal information required to perform SEEs detection?*

First, the validation criterion that must be satisfied by an algorithm to be defined as efficient are given in section 6.1. Then supervised detection is tested in section 6.2. In that case, it is assumed that both normal and abnormal observations are available, as well as their labels. After that, supervised detection boosted by expert opinion is tested in section 6.3. In this experiment, it is assumed the availability of both normal and abnormal observations, but without labels. Finally, one-class classification is tested in section 6.4. In that experiment, only normal observations are available during training. The exact setup of each experiment is displayed in table 6.1.

Table 6.1: Case study steps

Classes available	Labels available	Machine learning categories
Normal + anomaly	Normal + anomaly	Supervised learning (Classification)
Normal + anomaly	None	Classification boosted by expert opinion (Clustering + classification)
Normal	Normal	One-class classification

6.1 Validation criterion

There is no real consensus on what is a perfect result regarding machine learning algorithms. Nevertheless, in order to analyse their efficiency, it is needed to quantify the model's quality, and compare it to fixed criteria. Thus, the indicators previously detailed in table 3.1 are used to create three validation criteria C_1, C_2 and C_3, that will be used throughout this chapter. These criteria are chosen with respect to space industry standards for of single event effects detection.

- $C_1 : \ TPR_{test_D} = 100\%$.
  The detection must perform as good as the baseline detection method. In other words, it must be able to detect without fail all destructive anomalies. Thus, the true positive rate for the destructive test set must be 100%.

- $C_2 :\ ACC_{test_{ND}} \geq 85\%$.
  The detection must perform reasonably well regarding non-destructive anomalies. It can be tricky to define a threshold at which a model is defined as effective or not. Indeed, this value is heavily application dependant. For anomaly detection purposes, with only normal and abnormal classes, it is possible to say that a model is underperforming when its accuracy is below 50%. However, it is decided to set a higher threshold, so any model with an accuracy falling below 85% is rejected.

- $C_3 :\ min(TNR_{test_D} \geq 85\%,\ TNR_{test_{ND}} \geq 85\%)$.
  The detection must limit the abundance of false positives. Indeed, a false positive induces a power reboot of the monitored component, thus reducing utilisation time. If not actively limited, it would render the component useless for the mission. Therefore, a special attention is given to the true negative rate for all tests. As for criterion C_2, it is complicated to properly estimate a threshold regarding true negative rate. However, today's baseline detection method does not have false positives. In that case, even though 100% is not required, a strict 85% true negative rate is required to validate a model. To take into account both destructive and non-destructive anomalies, the lowest true negative rate is selected.

6.2 Supervised anomaly detection

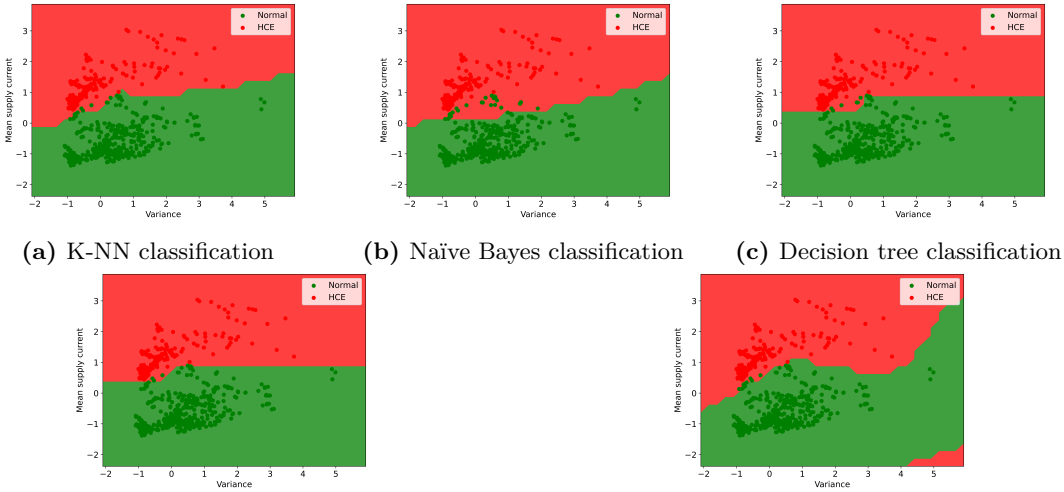
6.2.1 Principle

Supervised learning assumes the availability of both normal and abnormal labels. It builds a model based given labelled observations. By doing so, a separation of the classes is proposed by the classifier. It is then possible to predict whether a future sample is either normal or anomalous.

A selection of classification algorithms is tested to get an overview of the possibilities given by machine learning. The algorithms used are K-nearest neighbours [71], decision trees [72], Random forests [60], and support vector machines [58]. These models are chosen as they are the most commonly cited for classification problems [98–100]. An example of how each model divides normal mode from failure mode based on the same training set is given in figure 6.1.

6.2.2 Parameter selection

Parameter selection contributes extensively to the performance of a model. Consequently, many efforts are deployed in order to select optimal parameters to give a fair analysis regarding anomaly detection results.



(a) K-NN classification **(b)** Naïve Bayes classification **(c)** Decision tree classification

(d) Random forest classification **(e)** SVM classification

Figure 6.1: Classification examples

6.2.2.1 Hold-out validation

Two experiments are performed. First, the best parameters are set in order to detect destructive single event effects using $Train_{simD}$ database. Then another experiment is conducted using $Train_{simD}$ database in order to detect small single event effects.

In this study, only the training databases are used. The *hold-out validation* method is used to divide the data sets into a train set and a *validation set*. The difference between a test set and a validation set is that the validation set is used for parameter selection, while the test set is used to evaluate the model's performance. Moreover, unlike the test set, the validation set does not have to be decorrelated with the training set.

Algorithm 1: Supervised models parameter selection

input: $X = \{X_1, X_2, ..., X_n\}$ (set of time series $X_i = \{x_t, y_t, t \in T\}$)
$P = \{P_1, P_2, ..., P_m\}$ (set of parameters set)
output: Confusion matrix cm
Initialise the confusion matrix cm ;
foreach *data sets* $X_i \in X$ **do**
  Split X_i in two new data sets X_1 and X_2 ;
  foreach *sets of parameters* $P_j \in P$ **do**
    Train a classifier C using X_1 and P_j ;
    Make a prediction set $\tilde{y}_t$ of X_2 using the classifier C ;
    Update cm by comparing $y_t \in X_2$ and $\tilde{y}_t$;
  end
end
return cm ;

The hold-out validation method works as follows. A data set of size n is divided into two non-overlapping parts, the training set and the validation set [101]. Usually, the training set contains at least 60% of n. Cross-validation is another popular method to test machine learning models, which includes algorithms such as *K-fold* [102] or *leave one out* [103].

The whole parameter selection process is described in algorithm 1. First, the hold-out method is applied to the input data set, giving a train set X_1 and a validation set X_2 (line 5). The data set is either $Train_{simD}$ or $Train_{simND}$. Then, a classifier is trained on X_1 using a set P_j of parameters (line 7). Afterwards, The confusion matrix of this model is computed (lines 8 and 9). Doing so for each set of parameters P_j, it is then possible to compare the results of each model to decide the best set of parameters to be tuned.

6.2.2.2 Selected parameters

The number of neighbours N as well as the distance metric for K-NN, the number of trees N_{tree} for random forest and the kernel used in SVM are the parameters put to the test. The resulting parameters chosen after evaluation are displayed in table 6.2

Table 6.2: Parameter selection for supervised algorithms

	$Train_{simD}$	$Train_{simND}$
K-NN N	9	20
K-NN *metric*	Euclidean	Euclidean
Random Forest N_{tree}	10	10
SVM *kernel*	RBF	RBF

6.2.3 Results

The results of supervised learning for single event effects detection are discussed in this section. The analysis of the validation criteria is available in table 6.3 for all tested methods. In addition, the complete confusion matrix is available in table C.1 in appendices for an in-depth verification of the results.

All methods have proved efficient in detecting of single event effects, as all methods validated all three criteria. In regards to criterion C_1, all destructive single event effects have been correctly predicted by all algorithms. It means that supervised learning is as efficient as the baseline detection method regarding this task.
Furthermore, regarding criteria C_2 and C_3, the results are heavily outperforming. With an average accuracy for the non-destructive test set of 93.5%, and an average TNR of 92.7%, supervised methods are definitely a valid option for SEEs detection. Looking in detail at all methods shows that the decision tree algorithm is the least efficient method. Nevertheless, its performance is around 90% for both accuracy and true negative rate, which still validates all criteria. On the other hand, support vector machines outclassed other methods. Consequently, SVM would be considered the primary algorithm for this problem, assuming that both normal mode and failure observations are available and

the user has access to all labels.

These experiments demonstrated that supervised learning can be considered a viable alternative for detecting single event effects. However, this alternative can only be used in perfect training conditions (observations of failure mode and labels available during training), which are not necessarily fulfilled for space missions. Thus, other experiments are required to expand the analysis to other categories of machine learning algorithms.

Table 6.3: Results of supervised detection regarding the three criteria

	C_1 ($TPR_{test_D} = 100\%$)	C_2 ($ACC_{test_{ND}} \geq 85\%$)	C_3 ($TNR_{test_{D\&ND}} \geq 85\%$)
K-NN	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 93.95\%$ ✓	$TNR_{test_{simND}} = 92.92\%$ ✓
Naive Bayes	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 94.40\%$ ✓	$TNR_{test_{simND}} = 93.32\%$ ✓
Decision tree	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 90.58\%$ ✓	$TNR_{test_{simND}} = 89.22\%$ ✓
Random forest	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 94.42\%$ ✓	$TNR_{test_{simND}} = 93.60\%$ ✓
SVM	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 95.63\%$ ✓	$TNR_{test_{simND}} = 95.22\%$ ✓

6.3 Classification boosted by expert opinion

6.3.1 Principle

In this experiment, the labels have been removed from the training set. In that case, during the training phase, the data set is still composed of observations coming from both normal and failure mode, but the labels are not available to the algorithm.

In these conditions, new labels have to be created beforehand of the supervised detection. This part is performed by a combination of unsupervised clustering and *expert labelling*. The expert labelling method consists of using an expert to assess in which class falls each observation. However, treating each sample of all training sets would be too intricate. Thus, clustering is used beforehand to group samples based on similarity criteria.

Classification boosted by clustering testing methodology is described in algorithm 2. Note that the labels y_t are not available during the training phase. However, the labels of the testing sets are available in order to be able to evaluate the performance of the algorithms regarding the three criterion.
First, clustering is applied on the training set. It results in a division of the observations into clusters (line 5). Then the expert opinion is used to assign the different clusters to the most likely class (line 6). By doing so, it is possible to associate each observation x_t

of the training set to a label y_t. In the case of single event effects detection, it would be either normal or failure class. Afterwards, it is possible to evaluate the model based on the criterion. A classifier is trained on the training set augmented by expert opinion, then predictions are made using the test set to update the confusion matrix.

Algorithm 2: Classification boosted by clustering

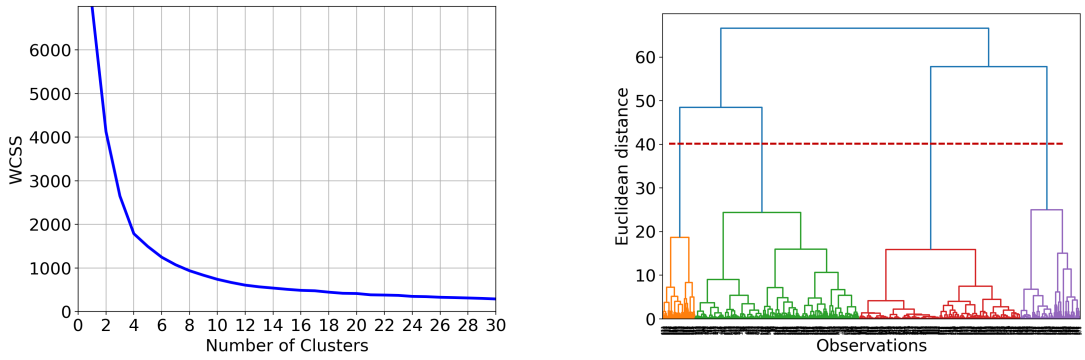
input: $X = \{X_1, X_2, ..., X_n\}$ (set of train sets $X_i = \{x_t, t \in T\}$)
$Z = \{Z_1, Z_2, ..., Z_m\}$ (set of test sets $Z_j = \{x_t, y_t, t \in T\}$)
output: confusion matrix cm
Initialise the confusion matrix cm ;
foreach *Training set* $X_i \in X$ **do**
  Apply clustering on X_i, resulting in the set of clusters $K = \{K_1, K_2, ..., K_n\}$;
  Apply expert labelling on K, resulting in the label set $y_t^K = \{y_1, y_2, ..., y_m\}$ (with $m \leq n$) ;
  Train a classifier C_i on X_i using the label set y_t^K ;
  foreach *test sets* $Z_j \in Z$ **do**
    Make a prediction set $\tilde{y}_t$ of Z_j using the classifier C ;
    Update cm by comparing $y_t \in Z_j$ and $\tilde{y}_t$;
  end
end
return cm

The main drawback of this method is that it multiplies prediction errors of both classification and clustering algorithms. However, the benefits are that the accuracy results are a relevant estimation of the performance of unsupervised learning techniques in the detection of anomalies. In addition, as we have seen in section 6.2.3, the classification performance is sufficient to have only a small impact on the prediction.
Four clustering algorithms are tested, including K-means [74], Hierarchical clustering [76], DBSCAN [77] and DyClee [78]. The support vector machines method is used for the supervised step, as it gave the best results in section 6.2.

Note that these methods performed offline training using a specific training set. However, some unsupervised methods rely only on incoming observations to perform both training and anomaly detection [104–106]. These methods were not considered in this work as it is not possible to guarantee that no anomalies are present at the beginning of the algorithm. Because single event effects are permanent anomalies, the algorithm would train on anomalous behaviour, leading to a high rate of false negatives.

6.3.2 Parameter selection

As stated in section 3.3.2, clustering is the action of regrouping similar samples in groups called clusters. The number of clusters is sometimes a parameter required by the algorithm (K-means, hierarchical clustering). External indicators (such as the elbow



(a) K-means elbow method

(b) Hierarchical clustering dendrogram

Figure 6.2: Optimal K search

method [107] or a dendrogram) are needed to find the optimal number of clusters, .
Other methods (such as DBSCAN or DyClee) resolve around different parameters to define the number of clusters. Thus, the user does not decide the final number of clusters. The parameter selection method largely varies depending on this factor, so these two cases are treated separately.

6.3.2.1 Number of clusters as a parameter

This section focuses on the parameter selection of K-Means and Hierarchical clustering methods. In these methods, the most important parameter is the number of clusters K to form from the training set. In order to find the optimal K value, different methods are available depending on the algorithm. The elbow method is used for K-Means (see figure 6.2a), and a dendrogram is used for hierarchical clustering (see figure 6.2b). Note that these figures have been created for each training set of the training database, but as they are similar for all data sets, only one example of each is shown here.

The results are noted in table 6.4 Considering K-means, the elbow method gives a clear indication of fracture from 3 clusters to 7 clusters. Thus, for K-means, it is possible to get $K \in [3, 7]$. In this experiment, $K = 4$.
Regarding hierarchical clustering, horizontally crossing by leaving the maximum height possible as shown in figure 6.2b gives an intersection of four clusters. In consequence, the number of clusters chosen for hierarchical clustering is also $K = 4$.

6.3.2.2 Number of clusters left to the algorithm

This section aims to find optimal parameters for clustering algorithms that do not consider formed clusters K as parameter. As the aim of this step is to facilitate the expert labelling step, a large number of clusters would be counterproductive. Therefore, an indicator was created during this thesis project in order to limit the number of clusters decided by these methods.

A *score function* is implemented (see eq.6.1) to help decide optimal parameters. It

is used on algorithms compute the final number of clusters during training. A penalty is given when the number of clusters exceeds a set threshold.

$$Score = \begin{cases} Accuracy & \text{if } \frac{K}{N} \leqslant R \\ Accuracy.e^{-(\frac{K}{N}-R)^{\frac{1}{P}}} & \text{if } \frac{K}{N} > R \end{cases} \tag{6.1}$$

with P the penalty, K the clusters created, N the true classes and R the ratio.

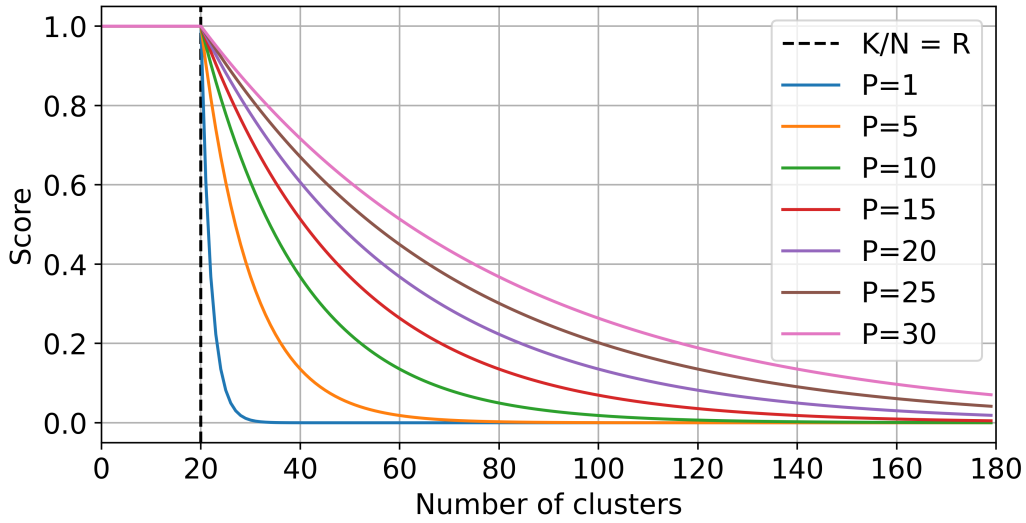


Figure 6.3: Score function for N=2, R=10 and accuracy=1

An example of how the score behaves depending on the number of clusters is given in figure 6.3. The values to be selected by the user are the Penalty P and the ratio R. P influences the weight of the penalty when there is a big number of clusters. The greater is P, the slower is the decrease of the score. R establishes the clusters ratio threshold when the penalty is applied. Using this score, we are able to select more efficiently the parameters of DBSCAN and DyClee to limit the number of clusters.

Table 6.4: Parameter selection for supervised algorithms

	$Train_{simD}$	$Train_{simND}$
K-Means K	4	4
Hierarchical clustering K	4	4
Hierarchical clustering *metric*	euclidean	euclidean
DBSCAN ε	0.05	0.01
DBSCAN *metric*	Manhattan	Euclidean
DyClee *g_size*	0.200	0.050

The chosen parameters are displayed in tab 6.4. The selection is based on the performance of a model regarding the score indicator. The epsilon (ε) value and the metric of DBSCAN and the g_size of DyClee are the parameters that are put to the test. Note

that in the table, the hierarchical clustering metric parameter has been tested using only the accuracy indicator.

6.3.3 Results

Table 6.5: Results of classification boosted by clustering regarding the three criteria

	C_1 ($TPR_{test_D} = 100\%$)	C_2 ($ACC_{test_{ND}} \geq 85\%$)	C_3 ($TNR_{test_{D\&ND}} \geq 85\%$)
K-Means	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 93.95\%$ ✓	$TNR_{test_{simND}} = 92.92\%$ ✓
Hierarchical clustering	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simD}} = 92.98\%$ ✓	$TNR_{test_{simD}} = 96.58\%$ ✓
DBSCAN	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 82.58\%$ ✗	$TNR_{test_{simD}} = 92.84\%$ ✓
DyClee	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 92.97\%$ ✓	$TNR_{test_{simD}} = 94.47\%$ ✓

The results on the three criteria in the case where the classification is boosted by clustering are available in table 6.5 for all tested methods. Also, the complete confusion matrix is available in table C.2 in the appendices.

Again, most of the methods have validated all three criteria. The best performing algorithm is hierarchical clustering in this experiment.
First, criteria C_1 has been validated for all methods, proving the capability of machine learning in the detection of destructive single event effects.
However, one method did not validate C_2. Indeed, DBSCAN accuracy for non-destructive faults falls below 85%. Looking at the in-depth results in table C.2, it appears that DBSCAN failed to detect all anomalies present in $Test_{simND}$. One hypothesis is that the density aspect of DBSCAN is not appropriate for this case study.
Finally, all algorithms validated criteria C_3. Overall, hierarchical clustering is the best performing unsupervised algorithm regarding the results obtained for all three criteria.

In conclusion, most algorithms proved to be efficient in the detection of single event effects. It demonstrates that even when no labels are available, it is still possible to accurately model single event effects by using machine learning algorithm. However, in a real case study, it is unlikely to be able to perform the training phase using single event effects observation. Therefore, the final step is to be able to prove the efficiency of machine learning limited to only normal observation during the training phase.

6.4 One-class classification

6.4.1 Principle

One-class classification regroups methods that try to model a system based on observations of a single target class. After training, the model is capable of identifying observations either as part of the target class or as outlier, as stated in section3.3.3. In the case of anomaly detection, it is possible to identify anomalies, such as single event effects, by training the model only on the system's normal behaviour. The goal is to demonstrate that one-class classification algorithms are efficient for the detection of single event effects.

This last experiment is crucial for the proof of concept. Indeed, proving that one-class classification algorithms are a valid approach for single event effect detection would avoid the need to perform extensive experiments to collect radiation fault examples.

The methodology used to analyse the algorithm's performance is the same as described in section 6.2.2. Note that only the $Train_{sim}$ database is used to train the model. The algorithms tested are Elliptic Envelope (EE), Local Outlier Factor (LOF) [82], Isolation Forest (IF) [81], One-Class SVM [58] and Auto-Encoders (AE) [83].

6.4.2 Parameters

As a sub-category derivated from classification, one-class classification parameter optimisation is treated similarly as in section 6.2.2. The hold-out method described in algorithm 1 is used to compare the performance of a batch of selected parameters for each method. At the exception that only normal class observations are retain to train the one class algorithm.
Note that during the parameter selection, labelled anomalous observations are introduced in the validation set to evaluate the selected parameters. It is done so the proof of concept is perform using optimal condition. However, it would not be possible to perform this step due to the lack of anomalous observations in real condition.

The tested parameters are the support fraction for elliptic envelope, the number of neighbours N to use during LOF calculations, the number of estimator M for isolation forest and the architecture (number of neurons per layer) for auto-encoders.
The chosen parameters are displayed in table 6.6.

Table 6.6: Parameter selection for supervised algorithms

	$Train_{simD}$	$Train_{simND}$
EE $supportfraction$	0.8	0.7
LOF N	10	30
IF M	40	100
OC-SVM $kernel$	RBF	RBF
AE $architecture$	64 32 16 16 32 64	64 32 16 16 32 64

6.4.3 Results

The results of the tests are displayed in table 6.7. Plus, the detailed confusion matrix is available in table C.3.

Regarding criterion C_1, only isolation forest is not able to identify all destructive single event effects. However, all other tested methods can perform equally with the baseline threshold method.
Focusing on criteria C_2 and C_3, Except auto-encoders, all methods validated them. An explanation of auto-encoders' underperformance might be that insufficient observations are available to train the model, as neural networks tend to need massive amounts of data to model a system correctly. Another explanation is that a lot of parameter tuning is required for neural network oriented methods. During this experiment, it was tried to give an equal amount of time to fine-tune each method as to avoid bias during the comparison. Therefore, it is possible that an in-depth study of auto-encoders parameters might improve the results.

In conclusion, three out of the five tested methods validated all three criteria. Therefore, it has been proven that one-class classification algorithms can be effectively used for the detection of single event effects. Consequently, it now possible to consider using only normal behaviour to train the model. Doing so removes the need to perform tedious experimental testing to emulate radiation fault observations. Rather than that, the system can be simply monitored and the resulting observations fed to the model.

Table 6.7: Results of one-class classification regarding the three criteria

	C_1 ($TPR_{test_D} = 100\%$)	C_2 ($ACC_{test_{ND}} \geq 85\%$)	C_3 ($TNR_{test_{D\&ND}} \geq 85\%$)
Elliptic Envelope	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 85.96\%$ ✓	$TNR_{test_{simND}} = 98.87\%$ ✓
Isolation Forest	$TPR_{test_{simD}} = 94.00\%$ ✗	$ACC_{test_{simND}} = 90.00\%$ ✓	$TNR_{test_{simD}} = 98.91\%$ ✓
LOF	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 88.87\%$ ✓	$TNR_{test_{simD}} = 91.44\%$ ✓
OC-SVM	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 94.11\%$ ✓	$TNR_{test_{simD}} = 95.20\%$ ✓
Auto Encoders	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 81.64\%$ ✗	$TNR_{test_{simD}} = 77.86\%$ ✗

6.5 Conclusion

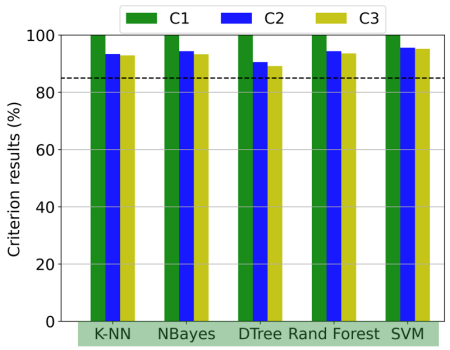
In this chapter, three experiments have been reported to establish the validity of machine learning for the detection of single event effects. First, three criteria have been stated to compare machine learning results with the baseline threshold detection method. Focusing

on both destructive and non-destructive anomalies, it is possible to judge the result of a detection method.

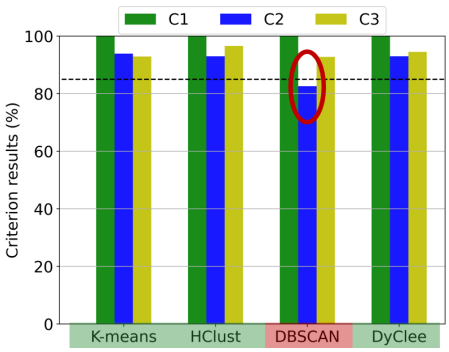
The three experiments resulted in the validation of anomaly detection methods for SEE detection. Indeed, out of the eleven selected algorithms, eight of them outperformed the baseline detection methods regarding destructive anomalies, while giving satisfactory results on non-destructive anomalies. The results of each category are summarised in figure 6.4

By looking at the specificities of each test, it appears that supervised detection methods are the most efficient at detecting single event effects. Nevertheless, three one-class classification algorithms validated all three criteria, even though only normal observations were available during training. This result indicates that depending on the availability of the observations during training, one can always select an efficient algorithm to detect SEEs.

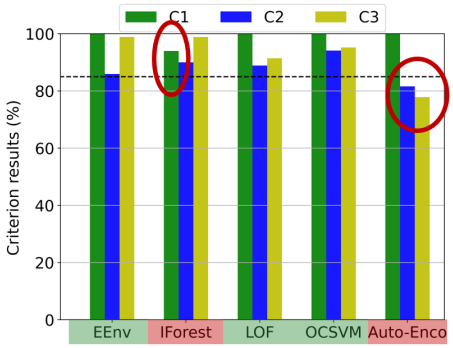
Nevertheless, space applications require specifications that are not always met by the algorithms selected in this study. In fact, few algorithms could be used on-board for a space mission. Therefore, the next chapter presents our proposal of a new anomaly detection method tailored to the requirements of space missions.



(a) Supervised anomaly detection results



(b) Classification boosted by expert opinion results



(c) One-class classification results

Figure 6.4: Classification examples

Chapter 7

Dynamic double Anomaly Detection DyD^2

Contents

Throughout the previous chapter, it has been demonstrated that machine learning algorithms are well fitted for the detection of single event effects. However, performance is not the only criterion that must be considered when planning a space mission. Adaptability, on-board constraints and real-time applications are examples of other requirements. Unfortunately, most machine learning algorithms are not designed to suit all of these specifications.

It is why a new anomaly detection has been developed to meet all the specifications of space applications. Called *Dynamic Double anomaly Detection (DyD^2)*[1] [109], this algorithm is designed as an alternative to one-class classification methods tested before.

In this chapter, the focus is on the description of the DyD^{2}algorithm. First, in section 7.1, the requirements fulfilled by DyD^2 are stated. Then in section 7.2, a formalisation of the anomalies considered by DyD^2 is proposed. After that, a description of the concepts underlying DyD^2 is done in section 7.3. In section 7.4, the DyD^2 algorithm is

[1]The code of DyD^2 is available on a GitHub repository [108]

extensively detailed. Following, the parameters that can be configured by the user are described in section 7.5. Finally, the complexity of DyD^2 is analysed in section 7.6.

7.1 Space specifications

Space applications impose many constraints that must be taken into account when planning a mission. Therefore, it is crucial to define the requirements that set the functionalities required when designing a new machine learning algorithm for the detection of single event effects in the case of an on-board space mission. In consequence, DyD^2 is designed to detect single effects in space missions while satisfying the requirements below:

7.1.1 Change point anomaly detection in time series data streams

The algorithm must be able to detect anomalies in data sets that take the form of time series data streams. Those anomalies are identified as anomalous change points in the time series. In space electronics, anomalies always manifest as high current events in the supply current. Moreover, the detection algorithm should be able to detect destructive SEE, such as single event latch-ups, without fail. Finally, it must be able to detect soft errors that can pass through the baseline threshold detection method.
DyD^2 is specifically designed to handle time series data stream, as each observations can impact future prediction. Furthermore, DyD^2 integrates a change point detection step followed by a two-phases anomaly detection.

7.1.2 Low memory usage

The algorithm must be able to be embedded and run on minimal resources. Indeed, some space missions are designed for microcontrollers with only a few Kbytes of flash memory. Therefore, the available memory space must be optimised as much as possible. DyD^2 is based on specific objects called μ-clusters that group together similar samples. Therefore, only the μ-clusters need to be stored in memory, instead of the entire data set, which drastically decreases memory usage.

7.1.3 Real-time detection

The algorithm must be able to run efficiently in real-time in regard to the capability of space components. With the reliability of the monitored component at stake, the algorithm must be able to process new observations as they arrive without delay.
DyD^2 uses a fast change point detection to decide whether new observations are potential anomalies. Doing so avoids processing non-anomalous data points, thus saving time. Also, DyD^2 is designed around the concept of μ-clusters, which accelerates the detection process. By doing so, fewer objects need to be addressed, as opposed to the entirety of the data set.

7.1.4 Adaptability in evolving environment

The algorithm must be able to adapt to a constantly changing environment. Indeed, due to the *total ionising dose*, components' behaviour evolve during the entirety of the mission. These evolutions modify the supply current significantly, so any training performed beforehand becomes irrelevant. Moreover, it can be complex to model the ageing process, as many factors influence it (e.g. type of component, process, internal structures) [89].
By the use of an update phase that constantly adapts to incoming data, DyD^2 is able to follow the deviation of incoming data and does not require additional training for that.

7.1.5 Training on normal behaviour only

The training phase must be performed using only normal data. Indeed, the emulation of single event effects can be tedious and complex, and removing the need for extensive radiation testing is crucial. For example, two main scientific hurdles emerged during our previous study [94]. First, it is challenging to characterise correctly the type of anomaly appearing during testing (such as SEL, SEGR, SEFI, SEU). Second, getting an extensive database of all possible anomalies for complex components, such as microcontrollers, is complicated. Therefore, the database can only be partially created, and so, the quality of the prediction of machine learning algorithms can be severely altered.
DyD^2 is part of a sub-field of machine learning called *one-class classification* [36, 80] in which only one type of observations is needed to create a model: those of normal behaviour.

7.1.6 Interpretability

The last requirement to consider is linked to the acceptability of the results of *black-box* models. The algorithm must be as interpretable as possible. It is well-known that it is complicated to determine precisely how deep learning algorithms give a specific prediction [110–115]. For the space industry, the possibility to interpret the prediction is essential to apply with confidence machine learning detection for radiation faults. It is why tools such as neural networks are not investigated in this work. There is no clear definition to evaluate the interpretability of a model yet, but some indicators can improve its comprehension.
DyD^2 is a deterministic algorithm. Its predictions are not based on probabilistic calculations. Also, the evaluation of an anomaly is designed around explicable tools such as the notion of reachability and μ-clusters. Finally, tools are developed to visualise the evolution of DyD^2 through a data set, giving the possibility to explain its predictions.

7.2 Change point detection

As stated in the previous section, DyD^2 is focused on detecting change point detection anomalies in time series data streams. There exists many applications requiring to detect change points, or ruptures in time series. Climate change detection, speech recognition or video analysis are all representative examples. Therefore, this problem is thoroughly studied, and many methods are developed in the literature [116–120]. The focus is on finding abrupt changes in data when the properties of the time series are modified. Let us consider a time series as in equation (7.1):

$$X = \{x_t, t \in T\} \tag{7.1}$$

with x_t an observation at the time t and $T \in \Re_+$ is the index set of X
Then, it is possible to define a *stationary time series* as a succession of events whose statistical properties are constant [121]. From there, a *change point* represents a transition between different events of the time series. The action of finding change points can be seen as a model selection problem [122] in which the aim is to find the best segmentation.

7.3 Principles of DyD^2

The DyD^2 algorithm is a real-time dynamic anomaly detection method designed to fulfil on-board requirements, particularly low computational cost. The goal is to efficiently detect several types of anomalies in multivariate time series data streams. The general idea is to first train a model offline with normal data. Then the data stream is checked against the model during the online detection phase. Taking inspiration from clustering techniques for data streams [78, 123], the model is composed of μ-clusters that group together data points according to a distance-based criterion. In DyD^2, this principle is used for two detection phases of the algorithm with different features to characterise data points:

- The first detection phase aims at detecting critical and heavily out-of-distribution anomalies. It is why it takes raw measured quantities as features. Such features do not require preprocessing, hence promoting speed. These features are called *outer features*, the model learned through this phase is called the *outer map*, and the detected anomalies are qualified as *outer anomalies*.

- The second detection phase aims at detecting subtle anomalies that require in-depth analysis. Relevant features are extracted in well-chosen time windows. These features are called *inner features*, the model learned during this phase is called the *inner map*, and the detected anomalies are qualified as *inner anomalies*. This phase is slower than the first phase and assumes that this type of anomaly is not time-critical.

As already mentioned, anomaly detection is approached as a one-class classification problem: only normal data are mandatory to train the outer and inner detection maps. Consequently, no extensive testing to gather faulty observations is required. However, the downside is that the user must ensure that no anomalies are present in the training set, as faulty behaviour could be learned, leading to false-negative results.

The DyD^2 algorithm manages objects called *samples* and *μ-clusters:*

Definition 1 *(Sample) A sample S is characterised by a couple (F_S, t_S) where F_S is a feature vector and t_S is a date.*

DyD^2 makes use of two types of samples. The *point sample* S_p is a vector of features coming from time series values of a given time t. The *window sample* S_w is a vector of features extracted from a window of time series values starting at a given time t.
In the following, the notation S is used when it applies indifferently to point sample or window sample.

Definition 2 *(μ-cluster) A μ-cluster μCl_k is defined by a characteristic vector CF_k of the following form:*

$$CF_k = (n_k, C_k, tc_k, tu_k) \tag{7.2}$$

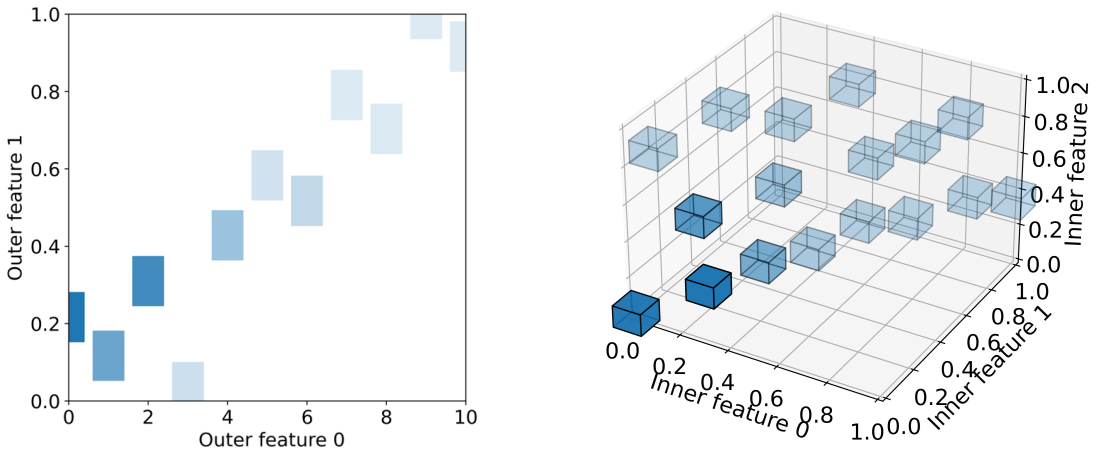
where $n_k \in \mathbb{N}$ is the number of samples in the μ-cluster, $C_k \in \mathbb{R}^+$ is a vector containing the coordinates of the μ-cluster center, $tc_k \in \mathbb{R}^+$ is the creation time of the μ-cluster, $tu_k \in \mathbb{R}^+$ is the time of the last update of the μ-cluster.

The initialisation of a μ-cluster is performed using a sample S. The feature vector of S is used as coordinates for the μ-cluster center C_k. Also, the date t of S is used for both tc_k and tu_k.

Definition 3 *(Detection map) A detection map M is composed of a set of learned μ-clusters of size $s_M \in (R^+)^m$, where $m \in \mathbb{N}$ is the dimension of the considered space. A detection map models normal behaviour.*

DyD^2 makes use of two detection maps. The *outer map*, denoted M_{out}, is created using point samples S_p. The *inner map*, denoted M_{in}, works with window samples S_w. Detection maps are dynamic objects in the sense that μ-cluster positions in the dimensional space defined by a set of features are adjusted depending on incoming data. Therefore, detection maps are able to follow data evolution that must not be considered anomalous (ageing or environmental modifications), hence allowing dynamic detection.

DyD^2 is developed as an on-board application, taking into consideration computing limitations. Low memory requirements and fast response time are key elements in on-board applications. The use of μ-clusters avoids the need to save all incoming data points by grouping them. By doing so, it is possible to work on significantly fewer



(a) outerMap **(b)** Inner map

Figure 7.1: DyD² maps

objects, thus saving computation time and memory space.

The notion of *reachability* [78] is used to perform the anomaly detection by evaluating the detection maps in regards to incoming observation.

Definition 4 *(Reachability) A μ-cluster μCl_k is reachable by a sample S if S is located inside the volume of μCl_k defined by its size s_k:*

$$distance(C_k, F_s) \preceq \frac{s_k}{2} \tag{7.3}$$

where $\preceq$ is the dimension-by-dimension $\leq$ relation.

By extension, a map M is reachable by a sample S if at least one μ-cluster of M is reachable by S. The Manhattan metric is used for the distance function as it performs better for high dimensions than the Euclidean distance [86].

In figure 7.1 is displayed an example of both outer an inner maps. The μ-cluster's distribution after training are displayed, with their respected density represented by the transparency. When a sample point is created, it is placed on the space displayed in figure 7.1a to see if it falls inside a μ-cluster. By doing so, it is possible to check the reachability of the point sample with the outer map. It is done similarly for the window sample and the inner map.

7.4 Algorithm

In this section, a thorough description of DyD² is done according to the flow chart in figure 7.5. The different steps are identified by circled numbers referenced in the corresponding sections and paragraphs.

In addition, a simplified flow chart is given in figure 7.2 in order to give an overview of DyD²'s algorithm. DyD² is divided into an offline training phase and an online detection.

The offline training phase ❶ is performed prior to the beginning of the mission. The user sets the hyper-parameters of DyD², such as the μ-clusters size, and the outer and inner maps are created using an anomaly-free training set.

The online phase is divided into four distinct steps. First, streaming data is processed with a change point detection ❷ that localizes potential anomalies in the stream. This analysis is critical to DyD² efficiency because it allows not to consider each sample as a possible anomaly and significantly improves the algorithm reaction time. To do so, the reachability is checked between a newly created point sample and a specific μ-cluster called *rupture μ-cluster*.
Then, a double anomaly detection ❸ is performed. As stated in section 7.3, this two-step detection allows to discriminate between destructive and non-destructive anomalies by the use of the outer and inner maps.
Finally, each observation predicted as normal is used to perform an update phase ❹ . A displacement of the μ-clusterscontained as well as an ageing process are applied to the detection maps. By doing so, it is possible to integrate deviation happening in the system.

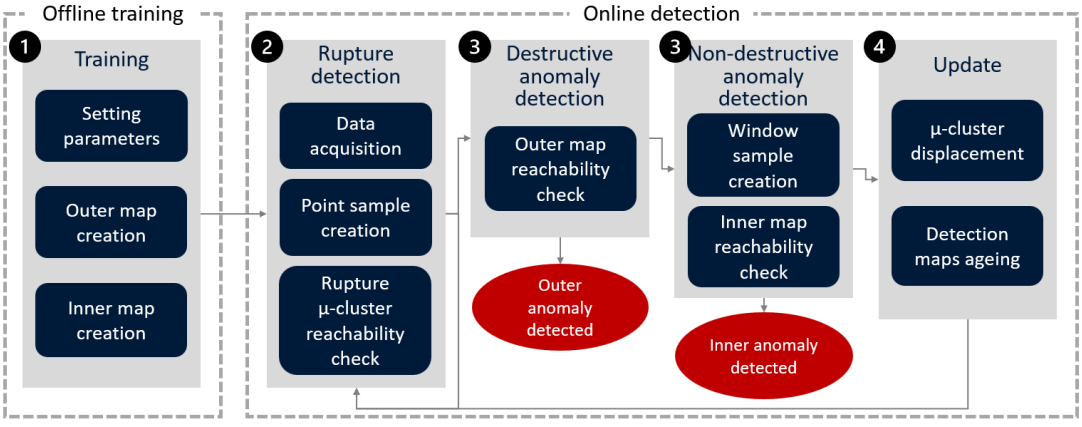


Figure 7.2: DyD² algorithm

7.4.1 Offline phase

7.4.1.1 Training ❶

DyD² training is performed offline. As a one class classification method, DyD² only needs normal data for the training phase. Therefore, the nominal behaviour of the

system is learned from historical normal data streams and stored in the detection maps M_{out} and M_{in}.
The training phase is detailed in Algorithm 3. First, a time window W of fixed size is defined (lines 3 and 4). W is used to create S_w and acts as a short-term memory that stores recent observations and moves through the data stream. For each new incoming observations, the first sample W_0 is erased (line 7). Given the incoming data stream, point samples S_p and window samples S_w are created (lines 8, 14, 15) to be checked for reachability against the μ-clusters of M_{out} and M_{in} respectively (lines 9, 16). If no μ-cluster of the detection map is reachable by the sample, a new μ-cluster is created using the sample characteristics (lines 10, 16). Otherwise, reachable μ-clusters are updated according to the sample feature vector (lines 12, 18).

Algorithm 3: Training

```
1  input: training set X = {x_t, t ∈ T}
2  output: inner map M_in and outer map M_out
3  initialisation of time window W = {w_1, w_2, ..., w_n} ;
4  for i ∈ [0, n[ do
5  |  w_i ← x_i ;
6  end
7  foreach observation x_i ∈ X (with i ≥ time window size) do
8  |  add x_i to w_i and remove w_1 ;
9  |  S_p ← (x_i, i) ;
10 |  if reachable(M_out, S_p) = false then
11 |  |  M_out ← initμCluster(M_out, S_p) ;
12 |  else
13 |  |  M_out ← update(M_out, S_p) ;
14 |  end
15 |  S_w ← featureExtraction(W) ;
16 |  if reachable(M_in, S_w) = false then
17 |  |  M_in ← initμCluster(M_in, S_w) ;
18 |  else
19 |  |  M_in ← update(M_in, S_w) ;
20 |  end
21 end
22 return M_out, M_in
```

An example of the training phase for a particular observation is displayed in figure 7.3. The training set as already been partially treated, and the focus is on the observation located around 420s. From this observation, a point sample S_p is created, as well as a window sample S_w with the use of a time window. From there, the reachability is verified between these two samples and their corresponding detection maps.
In this example, the outer map M_{out} is not reachable by the point sample S_p. In that case, a new μ-cluster is created in M_{out} at the location of S_p. On the other hand, the inner map M_{in} is reachable by the window sample S_w. In that case, the number of sample n_k of the k reachable cluster is updated.

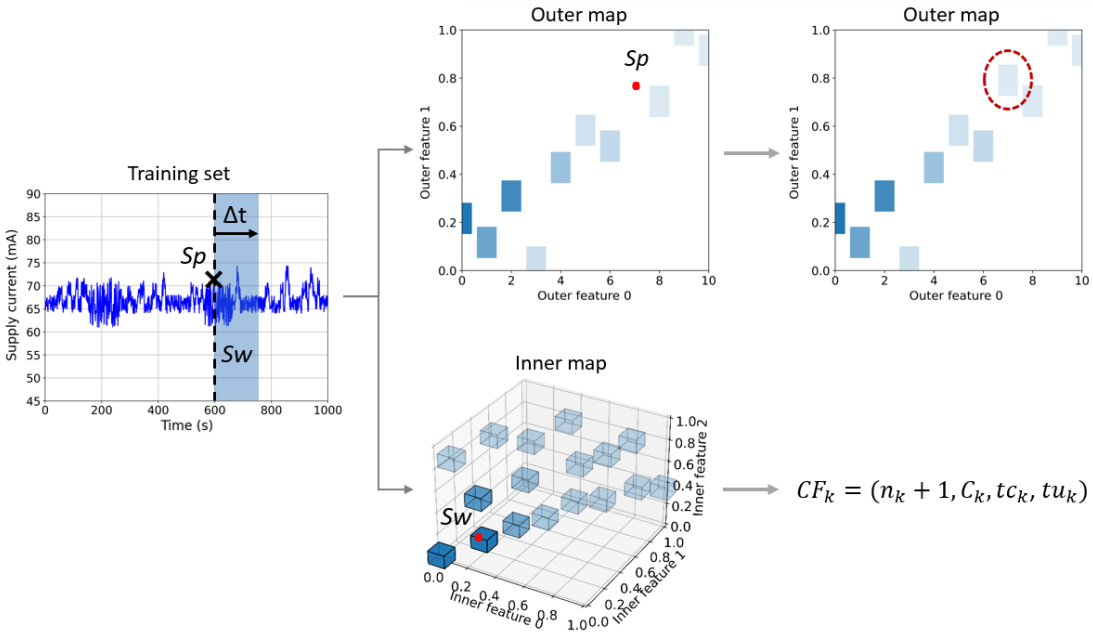


Figure 7.3: Training phase example

7.4.2 Online detection

When offline training is finalised and the two detection maps M_{in} and M_{out} have been created, DyD² can run on-board to detect anomalies on the fly based on the incoming data stream. As stated in figure 7.2, four steps are performed by DyD² during this phase. However, due to their similarities, the double detection phase is considered as one function

7.4.2.1 Change point detection ❷

In the considered application, anomalies are defined as change points. Hence, the first stage of DyD² is a change point detection to quickly exclude non-anomalous data from further processing, thus saving computation time. Algorithm 4 describes the change point detection method used in DyD².

Change point detection is performed using a single μ-cluster called *rupture μ-cluster* and denoted μCl_r. Unlike the μ-clusters of M_{out} and M_{in}, μCl_r is given a specific size $s_r \in (R^+)^{M_{out}}$ and is created during the online phase of the algorithm. The first μCl_r is created during an initialisation phase of the online detection (line 4). The first sample point S_p created during the online detection is used to create the first μCl_r. A change point is identified when μCl_r is not reachable by S_p (line 6). In that case, μCl_r is destroyed and a new rupture μ-cluster μCl_r is created using S_p (lines 7,8). Otherwise, μCl_r is updated using S_p.
Note that only one μCl_r is used throughout the lifetime of DyD². Indeed, when the current μCl_r is not used anymore, it is immediately replaced by a new one.

Algorithm 4: Change point detection

```
1  input: rupture μ-cluster μCl_r and point sample S_p
2  output: true if a change point is detected, false otherwise
3  if first iteration of the online phase then
4  |  initialise μCl_r with S_p ;
5  else
6  |  if reachable(μCl_r, S_p)=false then
7  |  |  destroy μCl_r ;
8  |  |  initialise new μCl_r using S_p ;
9  |  |  return true ;
10 |  else
11 |  |  update μCl_r using S_p ;
12 |  |  return false ;
13 |  end
14 end
```

7.4.2.2 Double anomaly detection ❸

In order to distinguish between critical from non-critical anomalies, DyD² performs two detection phases. For the first phase, the outer map M_{out} is used with outer features directly given by the coordinates of point samples S_p. This detection phase targets time-critical anomalies that are heavily out of the normal distribution. For the second phase, the inner map M_{in} is used along the window W from which a feature extraction process is performed to obtain window samples S_w. This feature extraction process is key to detect complex anomalies. The features to be extracted from the data window are left to the user, as they are application dependant.

Double anomaly detection is performed by checking S_p and conditionally S_w for reachability against the μ-clusters of the detection maps M_{out} and M_{in}, respectively. A critical *outer anomaly* is detected if M_{out} is not reachable by S_p. If this is not the case, detection goes on with S_w with respect to M_{in}. An *inner anomaly* is detected if M_{in} is not reachable by S_w.

7.4.2.3 Updating maps ❹

This step reflects the dynamic aspect of the DyD² method. It allows the integration of new knowledge about the current state of the system in the detection maps by updating M_{out} and M_{in} with samples that have been identified as normal. The update process is described in algorithm 5. In this process, the characteristic vectors of the μ-clusters contained in the detection maps are updated. Two steps are performed: center displacement and μ-cluster ageing

- *Center displacement* of μ-clusters is only performed on reachable μ-clusters in the map (line 3). Given a μ-cluster μCl_k, the displacement of its center is weighted by the number of samples n_k already present. The higher the amount of samples

Algorithm 5: Update map

```
1  input: Detection map M, sample S and ageing threshold τ
2  foreach k μ-clusters μCl_K ∈ M do
3      if μCl_k reachable by S then
4          foreach dimension m_k do
5              update C_k using eq (7.4) ;
6          end
7          n_k + + ;
8          tu_k ← t_S;
9      end
10     if tu_k ≥ τ then
11         Ageing using eq (7.5) ;
12     end
13 end
14 return M
```

inside μCl_k, the smaller the displacement due to the integration of a new sample S. The new μ-cluster center C'_k is given by the following formula to be understood as the center of mass:

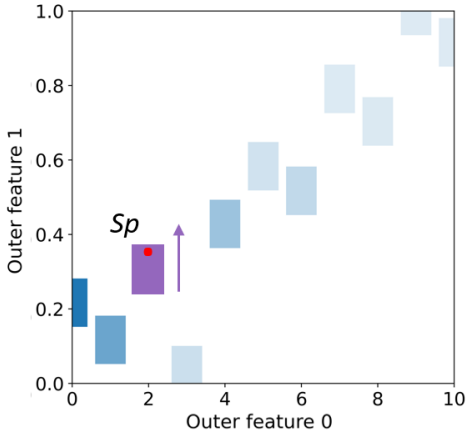
$$C'_k = \frac{S + n_k C_k}{1 + n_k} \tag{7.4}$$

μ-cluster centers update is performed in line 5.
The number of samples of the μ-cluster and the last update time are also updated (lines 7,8).
An example of center displacement for the outer and inner maps is dislayed in figure 7.4. On the left is shown the μ-clusters disposition before the update, while the μ-clusters disposition after the update is shown on the right. During the update, the reachable μ-cluster position (in purple) is modified accordingly to the sample coordinates.

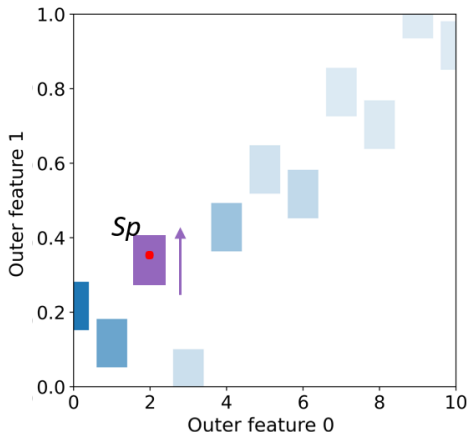
- *μ-clusters ageing* is performed to prioritise the most recent samples and forget those observed in the past. Indeed, mapping a dynamic behaviour means that newest information is more representative of the system's current behaviour. For a μ-cluster μCl_k of a map, the last update time tu_k is compared to the date of the last to arrive sample S (line 10). If the difference is higher than a fixed threshold, a penalty is applied to n_k (line 11). By doing so, old μ-clusters are the ones that will be the most impacted by future updates. The penalty is a linear decrease and is written as follows:

$$n'_k = n_k * penalty \tag{7.5}$$

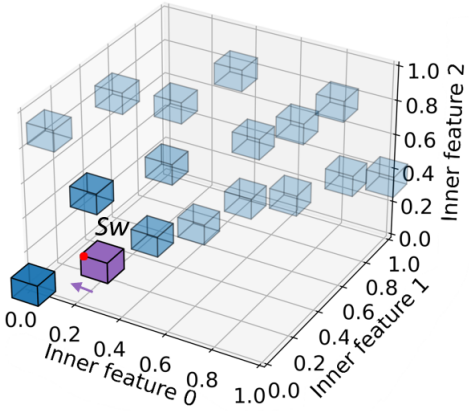
where n'_k is the new number of samples in μCl_k after applying the penalty.



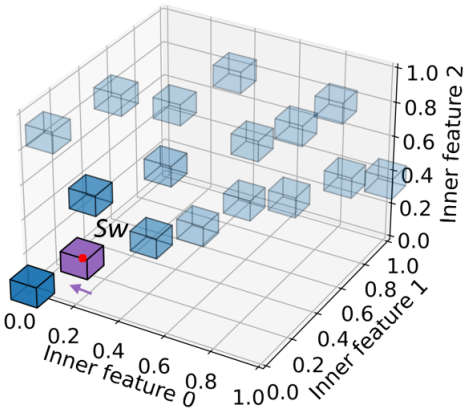
(a) Outer map before center displacement



(b) Outer map after center displacement



(c) Inner map before center displacement



(d) Inner map after center displacement

Figure 7.4: Center displacement during DyD² update

7.5 Parameters

The parameters required by DyD² are:

- **Rupture μ-cluster μCl_r size:** Defines the value at which a change point is detected. The higher this parameter, the less points are analysed by DyD², but the higher the risk of false negatives.
- **Outer and inner maps μ-cluster size:** Define the double detection accuracy. Small size means being able to detect more subtle faults, but it increases the risk of false positives. Note that it is possible to define a different μ-cluster size for each feature space dimension.
- **Time window size**: Defines the number of points used for creating the window samples and their features, which determines the inner map. A higher time window size means that more precise information can be given by the extracted features, but it also increases inner anomaly detection time.

- **Ageing threshold:** Defines the date limit at which a μ-cluster starts to decay. Decaying starts when the difference between the last update of a μ-cluster and the current date is greater than this parameter. A low value tends to favour dynamic behaviour, but it can increase the number of false positives as mapped behaviours may evolve too quickly.
- **Decay penalty:** Defines the decay hastiness of an old μ-cluster. It works as a decreasing linear function on the μ-cluster number of samples n_k (see eq. 7.5).

7.6 Complexity of DyD^2

The complexity analysis of an algorithm starts from a simple question asked by all computer scientists: "How long running this algorithm will take and how much memory do I need?". If all users used the exact same system, it would be simply answered by *benchmarking*. Benchmarking is running the algorithm while measuring time and memory consumption. However, with the large variety of existing systems, other methods have to be used.

Complexity analysis is crucial for embedded systems, as it gives information of an algorithm behaviour in terms of computation time and memory usage. Therefore, the the complexity of DyD^2 is the focus of this section. The big-O $O(n)$ notation [124] is used to evaluate the worst-case complexity of DyD^2. Two distinct evaluations are needed to estimate the complexity of DyD^2. Each evaluation refers to one aspect of DyD^2: the offline training and the online detection. In order to get a grasp of the complexity of DyD^2, these parts have to be studied separately.

7.6.1 Offline training complexity

7.6.1.1 Space complexity

To calculate the space complexity of the training phase, it is important to look at the memory created during the whole training process. The main objects manipulated by DyD^2 are samples and μ-clusters. It is important to assess that both of these objects are constant in memory. Indeed, their size is constant regardless of the input used to create such objects. Even though the space required depends on the number of features fixed by the user, it stays the same during the execution of the training phase. The input considered is the observations contained in the training set.

Let us dive inside a single loop of the algorithm.
First, a time window is initialised to be used for the creation of inner samples. The size of the window remains constant throughout the training phase, therefore the complexity is O(1). Then, outer samples and inner samples are created. These samples are needed for only one iteration of the loop and can be erased at the end of each iteration. Again, as neither the number of outer samples nor inner samples are evolving, the complexity is O(1). Finally, the core concept of the training phase is the creation of μ-clusters to model the normal behaviour of a system. Therefore, if the current sample is different from

previous observations, a new μ-cluster is created. The user can influence the number of μ-clusters created by adjusting the μ-cluster size of a detection map. However, unlike algorithms like K-means or hierarchical clustering, it is not possible to set a fixed amount of clusters to be discovered. Let us study the worst case:
Let us take a training set $X = x_1, x_2, ..., x_n$. If for each point (x_i, x_j) such as $dist(x_i, x_j) > s_k$, with s_k the size of all μ-clusters inside a detection map, then the final number of μ-clusters N is $N = n$.
The conclusion is that in the worst-case scenario, the number of μ-clusters is the same as the number of observations in the training set. A linear relation can be established, and because a μ-cluster is a constant object, it is possible to conclude that the creation of μ-clusters is of linear O(n) complexity.

In conclusion, only μ-clusters creation impacts space complexity during the training phase. Because μ-clusters creation is O(n), the overall space complexity of offline training of DyD^2 is O(n). It is important to understand the consequences of this result. Indeed, DyD^2 is designed to work in low memory environments. Therefore, the higher the number of μ-clusters the more memory space is needed to run the algorithm. Thus, two goals have to be achieved during the offline training phase of DyD^2. The detection map has to model as precisely as possible the system, while assuring that the number of μ-clusters created is not overflowing the memory. The user has to achieve a balance between precision and memory space by adjusting the map size parameter as well as the number of observations in the training set.

7.6.1.2 Time complexity

As for the space complexity, the time complexity is calculated by evaluating the complexity of a single observation. Again, the inputs correspond to the observations contained in the training set.
First of all, the creation and update of the time window, as well as the creation of samples can be seen as an assignment. Therefore, the time needed to perform this step is constant O(1). The interesting part is, again, the management of μ-clusters. As stated during space complexity evaluation, it is possible to discriminate the worst case of the offline training phase by the creation of μ-clusters for each observation of the training set. To update the detection map, it is needed to check the sample created by the latest observation with all μ-clusters contained in the detection map. In other words, for each observation, it is needed to go through a list of objects that is as long as the indices of given observations. This kind of algorithm is of O(n^2) complexity as it involves a loop contained in another loop.

In the end, the highest complexity is within the creation and update of μ-clusters. Therefore, the complexity of the offline training phase of DyD^2 is O(n^2).

7.6.2 Online detection complexity

7.6.2.1 Space complexity

For the most part, the space complexity of the online detection phase is very similar to the one of the offline training. Indeed, the time window and the sample creation mechanism are similar, thus the complexity is O(1). The key difference between the offline training phase and the online detection phase is the management of the detection maps. When a sample is not reachable by any μ-clusters that constitutes a detection map, the point is considered as an anomaly and DyD^2 is put on hold. Unlike the training phase, no μ-cluster is created. Therefore, the number of μ-clusters contained in a detection map remains constant throughout the algorithm and the complexity is O(1).

As a result, the overall space complexity of the offline training phase is constant O(1). It is crucial information for any user wanting to design an embedded algorithm. Indeed, the memory staying constant throughout the mission means that the risk of overflowing the memory at runtime is inexistent.

7.6.2.2 Time complexity

Regarding the time complexity for the online phase, it would be possible to tackle the problem with two different approaches. First, it is possible to reason similarly to the space complexity, and express the time complexity based on the input size n. In that case, the same conclusion can be applied by comparing the offline and online phase, and it is deduced that the time complexity is only dependant of the input size, as the number of μ-clusters remains constant throughout the execution of DyD^2. Therefore, the online phase time complexity is O(n).
Second, due to the online phase being a real-time application, an alternative would be to analyse the time complexity for a single observation. In that case, a possible input would be the number of μ-clusters m in the outer and inner detection maps. By taking the worst possible case, it can be considered that both maps have to be checked for each observations. The reachability of a sample is verified for all μ-clusters of a detection maps. Thus, the time complexity is linearly dependant of the total number of μ-clusters O(m).

The result is that the overall time complexity of the online detection phase is linearly dependant of the input O(n), while it is linearly dependant of the number of μ-clusters in the outer and inner detection maps O(m) when looking at a single input. Again, this result is critical for an embedded application. Indeed, not only the time needed to perform a detection loop is, in the worst possible case, constant, but it is also possible to impact the prediction time of an input by adjusting the number of μ-clusters during training.

7.7 Conclusion

In this chapter, an extensive description of DyD^2 has been reported. First, the space specifications that were set when designing DyD^2 are detailed. After that, a description of the core principles of DyD^2 is given. DyD^2 is based on the idea of moving μ-clusters that are grouped in detection maps. By checking the reachability of new observations, called samples, with the μ-clusters contained in a detection map, DyD^2 is able to differentiate between normal and abnormal observations.

From there, DyD^2 is working with two phases. The offline phase can be assimilated with training on normal observations, and the online phase can be assimilated as the prediction phase. The online phase works as a four stage algorithm. It starts with a change point detection phase, followed by two detection phases that discriminate between destructive and non-destructive anomalies, followed by an update phase that takes into consideration the system behaviour.

After that, the complexity of DyD^2 is provided. Divided between the offline and online phase, it is shown that time and space complexity of DyD^2 are suitable for real-time embedded applications.

Now that the theoretical background of DyD^2 has been laid, assessing its performance on single event effects detection is needed.

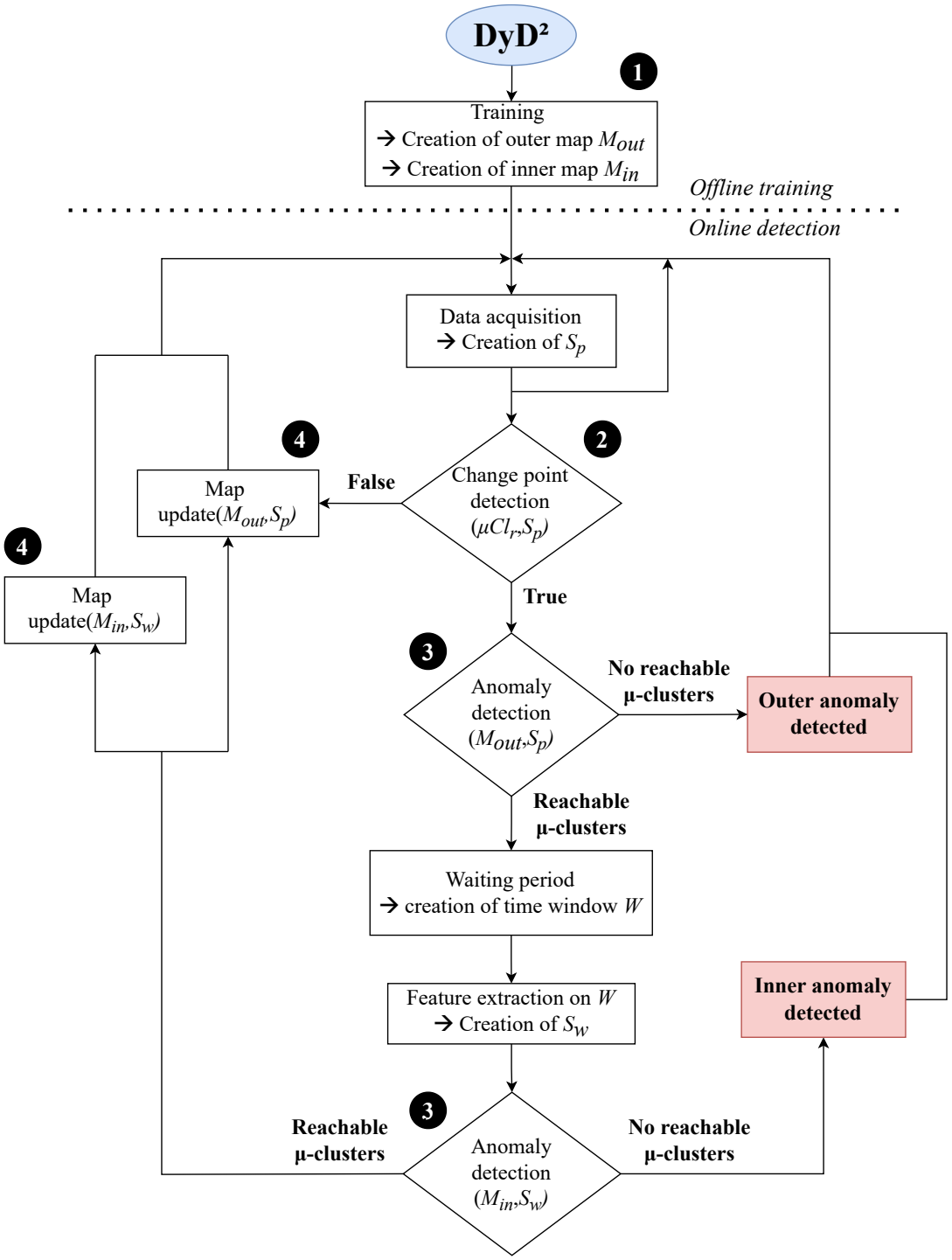


Figure 7.5: DyD² flow chart

Chapter 8

DyD^2 results

Contents

The Dynamic Double anomaly Detection (DyD^2) algorithm's performance on single event effects detection are put to the test in this chapter. Moreover, all aspects of DyD^2 must be evaluated. Consequently, in addition to the evaluation based on stationary data sets, DyD^2 is applied on dynamic systems to evaluate its adapting ability. Moreover, its capacity to perform in real-time on embedded applications must be assessed. To do so, experiments using the databases described in section 5.5 are performed. If DyD^2 validates all tests, it would validate one of the first machine learning algorithms for single event effects specifically designed to suit space applications.

This chapter resolves around three questions:

- *Is DyD^2 performance on par with state-of-the art methods?*
- *Is DyD^2 able to adapt to a dynamic environment?*
- *Is DyD^2 suited for real-time embedded applications?*

First, the performance of DyD^2 are evaluated on simulated data sets in section 8.1. A comparison with a selection of one-class classification methods is reported for both stationary and dynamic systems. Then DyD^2 is experimented on experimental data sets in section 8.1. An offline experiment is performed on observations gathered by heavy ion testing. Moreover, a laser experiment in a real embedded setup is reported.

8.1 DyD^2 results on simulation

The first test performed with DyD^2 is based on the data set created by computer simulation (see section 5.2). As for one-class classification experiments, DyD^2, is trained on data sets that only contain normal observations. Then, during testing, destructive and non-destructive single effects detection are treated separately. These are identified respectively as outer and inner anomalies regarding DyD^2 concepts. In addition, the dynamic behaviour of DyD^2 is also evaluated using simulated databases including a linear deviation throughout each test sets. It is important to note that this linear deviation is not considered as an anomaly in the system.

The databases used in each experiment were described in section 5.5. For each experiment, DyD^2 is trained with the $Train_{sim}$ simulated database. The first experiment is based on stationary data sets. Destructive and non-destructive anomaly detection are evaluated by using respectively $Test_{simD}$ and $Test_{simND}$ databases. Then, to evaluate DyD^2's capacity to adapt to a dynamic system, the simulated databases $Test_{simDevD}$ and $Test_{simDevND}$ are used. Again each of these couples contains respectively destructive and non-destructive anomalies, in addition to a linear deviation throughout each test set.

Finally, the three criteria introduced to analyse algorithm's performance in section 6.1 are used here to evaluate DyD^2 on simulated data sets.

8.1.1 Parameters

In this section, the parameters of DyD^2 for the series of experiments reported above are detailed. As DyD^2 is developed as an OCC algorithm, it is not possible to predict the type of anomalies encountered during the online detection phase. Consequently it is decided that the parameters chosen must remain unchanged for all experiments on simulated data. In doing so, the same conditions hold for all tests, and can give conclusive results regarding the efficiency of DyD^2 in detecting multiple types of anomalies.

In order to find optimal parameters, DyD^2 is executed on a batch of data sets. From there, the best results are taken. The parameters of DyD^2 (see section 7.5) are presented in table 8.1.

Table 8.1: DyD^2 parameters for tests on simulated data

μCl_r size	outer μCl size	inner μCl size	window size	age threshold	decay penalty
0.20	0.05	0.15	20ms	150ms	0.975

Table 8.2: Results of DyD^2 on simulated data regarding the three criteria

	C_1 ($TPR_{test_D} = 100\%$)	C_2 ($ACC_{test_{ND}} \geq 75\%$)	C_3 ($TNR_{test_{D\&ND}} \geq 85\%$)
DyD^2 Stationary test	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 93.55\%$ ✓	$TNR_{test_{simD}} = 93.97\%$ ✓
DyD^2 Dynamic test	$TPR_{test_{simD}} = 100\%$ ✓	$ACC_{test_{simND}} = 90.17\%$ ✓	$TNR_{test_{simND}} = 89.24\%$ ✓

8.1.2 Results

The results for both stationary and dynamic experiments are displayed in table 8.2. These results are going to be described in the two following subsections.

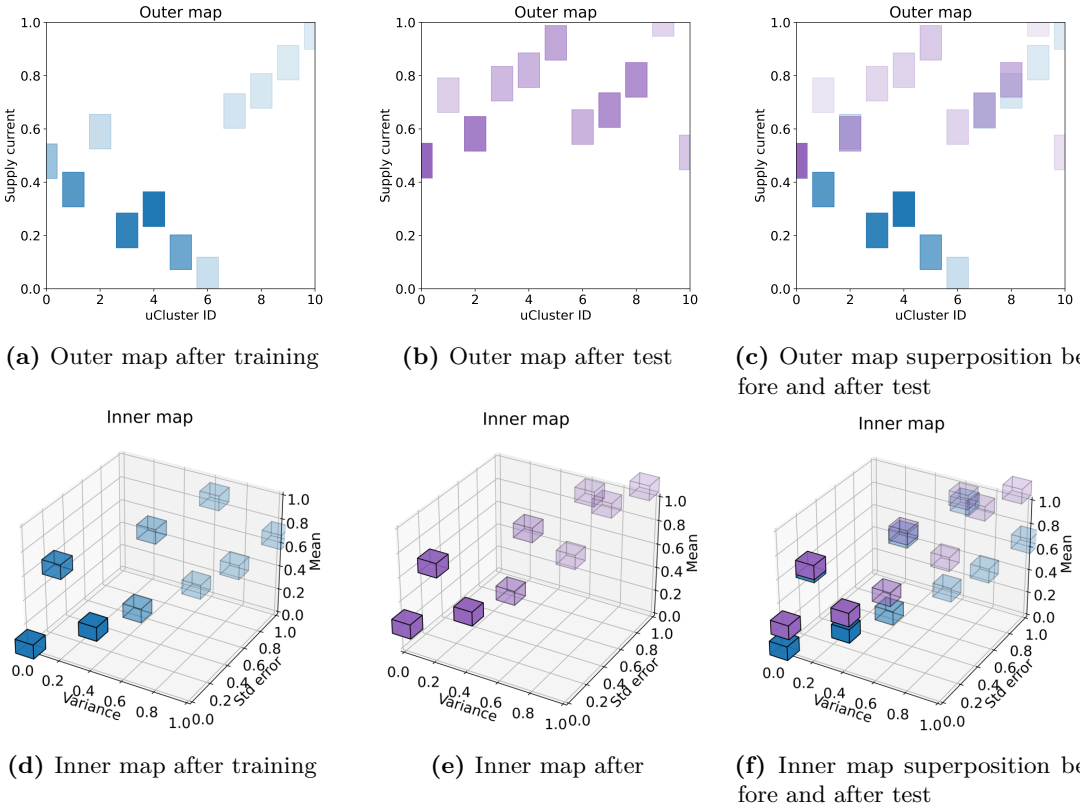
8.1.2.1 Stationary behaviour

The first line of table 8.2 gives the performance of DyD^2 regarding the three criteria on stationary data sets. Looking at C_1, one can notice that DyD^2 succeeded in detecting all anomalies in $Test_{simD}$. It indicates that DyD^2 performs as well as the baseline threshold protection by correctly identifying all destructive anomalies. Moreover, the accuracy is 93.5%, which validates the C_2 criteria. Lastly, the focus is on the false negative rate achieved by DyD^2. With a result of 94.6%, DyD^2 validates the last criterion.
Overall, DyD^2 performance match the performance of state-of-the-art anomaly detection methods. With the validation of all three criteria, DyD^2 can be confirmed as a valid method for the detection of single event effects.

8.1.2.2 Dynamic behaviour

DyD^2 is designed as an algorithm that is able to adapt to a dynamic environment. Thus, it is necessary to evaluate DyD^2 in this particular aspect. To do so, the databases $Test_{simDevD}$ and $Test_{simDevND}$ are used. In addition to representing respectively destructive and non-destructive single event effects respectively, they both feature a linear deviation on the data.

Again the same three criteria C_1, C_2 and C_3 are used to evaluate the performance of DyD^2 on dynamic data sets. This test's results are displayed on the second line of table 8.2. First, looking at criterion C_1, it is clear that DyD^2 can detect without fail all destructive faults occurring in $Test_{simDevD}$. Thus, by validating this first criterion, DyD^2 matches the detection performance of the baseline threshold-based detection method, even for data sets with dynamic behaviour. The same conclusion can be made for criterion C_2, as the accuracy of DyD^2 for $Test_{simDevND}$ is around 90.2%.
Finally, let us look at criterion C_3 which evaluates the true negative rate. Again, DyD^2 validates this criterion with a true negative rate of 89.2%. It means that DyD^2 correctly predicts most normal observations. Regarding this result, one can assess that DyD^2 succeeded in following the linear deviation present in $Test_{simDevND}$. However, by



(a) Outer map after training

(b) Outer map after test

(c) Outer map superposition before and after test

(d) Inner map after training

(e) Inner map after

(f) Inner map superposition before and after test

Figure 8.1: Example of inner map evolution after a test set including a linear deviation

comparing this test with the results obtained previously with the stationnary database $Test_{simND}$, it is possible to see that DyD^2 performance are lower for the dynamic test. During this experiment, it was observed that if the slope of the deviation is too important, DyD^2 needs time to re-adjust to the system by relocating its micro-clusters in the outer and inner detection maps.

The evolution of an outer and inner map is shown in figure 8.1. In figure 8.1a and 8.1d are depicted the detection maps after the training while in figure 8.1b and 8.1e are depicted the detection maps after testing, when the μ-clustershave evolved due to the observations contained in the test set. In figure 8.1c and 8.1f, the superposition of the maps before and after testing is displayed to emphasis the μ-cluster's movement. Regarding the outer map, most of the μ-clusters went up on the supply current axis. A similar conclusion can be made for the inner map, as most μ-clusters moved forward along the supply current mean axis. These outcomes agree with the linear deviation. Therefore, the μ-clusters' evolution results from DyD^2 adapting to the linear deviation present in the testing set.

8.1.2.3 Comparison with state-of-the-art one-class classification algorithms

As it can be complicated to evaluate the performance of a single machine learning algorithm, DyD^2 results on simulated data are compared with those of the one-class

classification algorithms used in section 6.4, namely Elliptic Envelope (EE), Local Outlier Factor (LOF) [82], Isolation Forest (IF) [81], One-Class SVM [58] and Auto-Encoders (AE) [83]. Doing so makes it possible to position DyD^2 with respect to the state-of-the-art methods in order to give an honest estimation of its performance.

The results of the selected OCC algorithms for the stationary experiments discussed in section 6.4 are used to compare DyD^2. In addition, further experiments have been made to evaluate OCC algorithms on dynamic data sets by using the databases $Test_{simDevD}$ and $Test_{simDevND}$. The results are available in table C.4 and in figure C.1 in appendices.

A summary of the tests is displayed in figure 8.2. Note that, as DyD^2 already satisfied all criterion, and to simplify the comparison between the algorithms, only the accuracy indicator and the execution time are used. These two indicators are available for DyD^2 and the five OCC algorithms. On the figure, the blue, green, red and purple bars represent the accuracy for the experiments performed on respectively $Test_{simD}$, $Test_{simND}$, and $Test_{simDevND}$. Lastly, the yellow bar represents the average time of each model to perform all four tests.

First, let us focus on the two stationary experiments. It is possible to see that DyD^2 performance are on par with state-of-the-art methods. Indeed, the accuracy of most algorithms is between 93% and 97.5% for both destructive and non-destructive anomalies. In this regard, DyD^2 can be considered a solid alternative for any of the selected OCC algorithms.
Secondly, let us dive into the results of the dynamic experiment. In this experiment, DyD^2 was still able to validate all three validation criteria. Looking at the results of the other five algorithms, it appears clear that they could not follow the linear deviation present in $Test_{simDevD}$ and $Test_{simDevND}$. Indeed, the accuracy dropped below 70% during the dynamic experiment. In conclusion, by outperforming the state-of-the-art algorithms on these databases, DyD^2 proved its capability to follow the deviations of a dynamic system.
Lastly, the yellow curve indicates the average time of each algorithm in the four experiments. DyD^2 fastest of all tested algorithms. It is possible to pinpoint the isolation forest algorithm, which took on average 38.70s, while the second slowest algorithm only took 1.71s on average. Also, the auto-encoders method is the second fastest algorithm, but it also possesses the lowest accuracy among all algorithms for all four experiments. In the end, these results comfort the efficiency of DyD^2 for real-time applications.

In conclusion, DyD^2 it is possible to say that DyD^2 is on par with the state-of-the-art methods regarding stationary systems. This result is significant because it validates DyD^2 as a functional one-class classification algorithm and assesses that DyD^2 can be used as an alternative to these well-known methods. Furthermore, DyD^2 proved its ability to adapt to dynamic behaviour as well as to provide a fast detection method when compared to other algorithms. Therefore, DyD^2 is proved to be an efficient solution for single event effects detection, not only in regard to today's threshold-based detection method, but also in regard to other state-of-the-art detection methods.

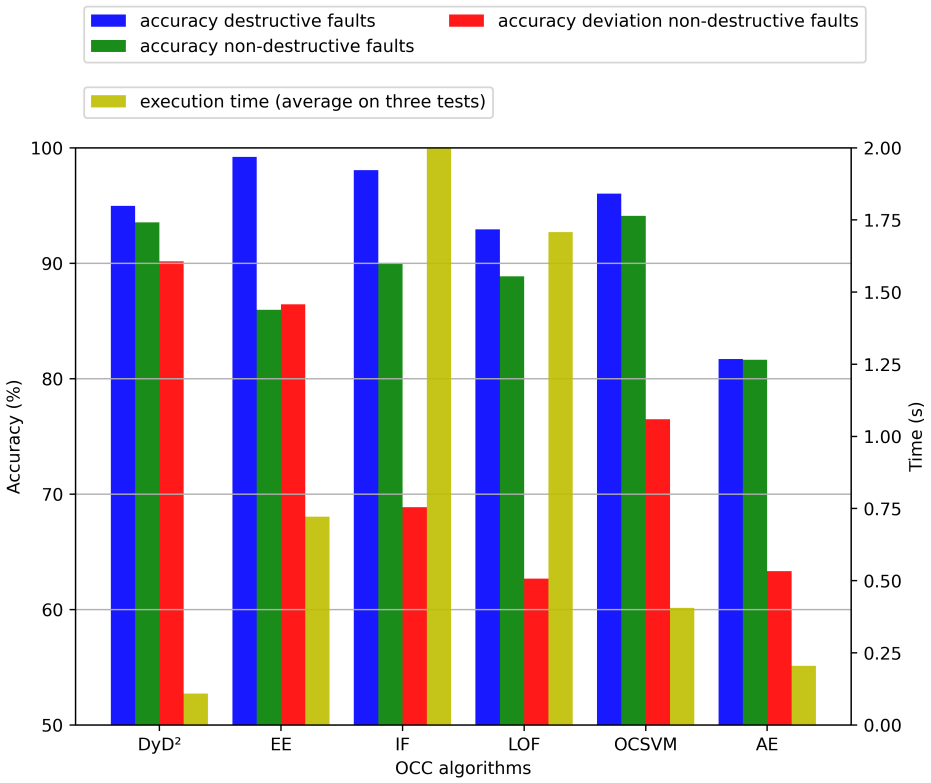


Figure 8.2: DyD² comparative results on simulated data sets

8.2 DyD² results on experimental tests

In the previous section, the performance of DyD² has been demonstrated using an extensive database created using simulation. However, simulation experiments alone are not conclusive evidence to validate DyD² for real case applications. DyD² has to be tested on data sets used by the space community as representative of satellite-based applications.

To do so, DyD² is applied on $train_{Hion}$ and $test_{Hion}$ databases. As a reminder, these data sets are coming from heavy ion testing performed by the CNES. These are highly representative of the damages caused by space radiations on electronic components and hence suitable to analyse the performance of DyD². In addition, online tests are performed on laser testing with DyD² executed on a SAM3X8E microcontroller. It is done to evaluate the capability of DyD² to perform in real time on a low power component.

8.2.1 Offline heavy-ion testing

DyD² is firstly trained on $train_{Hion}$, and then it is tested on $test_{Hion}$. The characteristics of these databases are available in section 5.5. The parameters used for this experiment are given in table 8.3.

Table 8.3: DyD^2 parameters for tests for heavy ion testing

μCl_r size	outer μCl size	inner μCl size	window size	age threshold	decay penalty
0.20	0.01	0.05	20ms	100s	0.975

The experimental setup for gathering these data sets do not allow to know precisely when a fault occurs. Therefore, only faults picked up by a threshold of 200mA can be considered. Consequently, even if a fault due to heavy ions occurred during the tests, but with a limited impact on the supply current, it cannot be labelled as an anomaly. Only the performance of destructive anomaly detection can hence be evaluated.
In this context, on the three validation criteria stated in section 6.1, only C1 and the indicator based on destructive anomalies for C_3 can be evaluated. The performance of DyD^2 regarding these two criteria are displayed in table 8.4, while detailed results are available in table C.4 in appendices. DyD^2 performance on the heavy-ion database are

Table 8.4: DyD^2 results on heavy-ion database

	C_1 ($TPR_{test_D} = 100\%$)	C_3 ($TNR_{test_{simD}} \geq 85\%$)
DyD^2	$TPR_{test_{Hion}} = 100\%$ ✓	$TNR_{test_{Hion}} = 99.55\%$ ✓

extremely positive. Indeed, not only every single anomalies detected by the threshold-based detection method are also detected by DyD^2, but also the rate of false detection is really low. Furthermore, it is possible to calculate the total time lost in false detection if DyD^2 was used in a real case application. Indeed, 50 minutes of recording time are totalised in $test_{Hion}$. As shown in table C.4, the false positive rate is equal to 0.45%. Therefore, only 13.5 seconds out of 50 minutes of active time is lost due to false detections.

In addition, an example of the detection result is given in figure 8.3. The supply current is represented by both blue and green dots. When the supply current is represented by a blue dot, it means that DyD^2 is active and looking for anomalies, while when the supply current is represented by a green dot, it means that DyD^2 detection is on hold, and is waiting for the supply current to return to a normal value. In this test, we consider that the system is back to its normal state when a new point sample reaches the outer map. The vertical red lines indicate when an outer anomaly is detected, and the vertical orange lines indicates when an outer anomaly is detected.
In this example, DyD^2 is able to detect all anomalies picked up by the threshold detector of 200mA. These anomalies are flagged as outer anomalies by DyD^2. In addition, three observations are flagged as inner anomalies by DyD^2 that fall below the detection threshold. Nevertheless, these observations clearly deviate from normal behaviour. In

these cases, DyD² is able to detect anomalies that are not picked up by the baseline detection method.
Furthermore, the capability to discriminate between destructive and non-destructive anomalies can be used to assist decision-making regarding the response when confronted with an anomaly. Indeed, while a fast and important response must be done regarding outer anomalies (such as power reset of the component), the response when confronting with a less critical inner anomaly can be adapted.

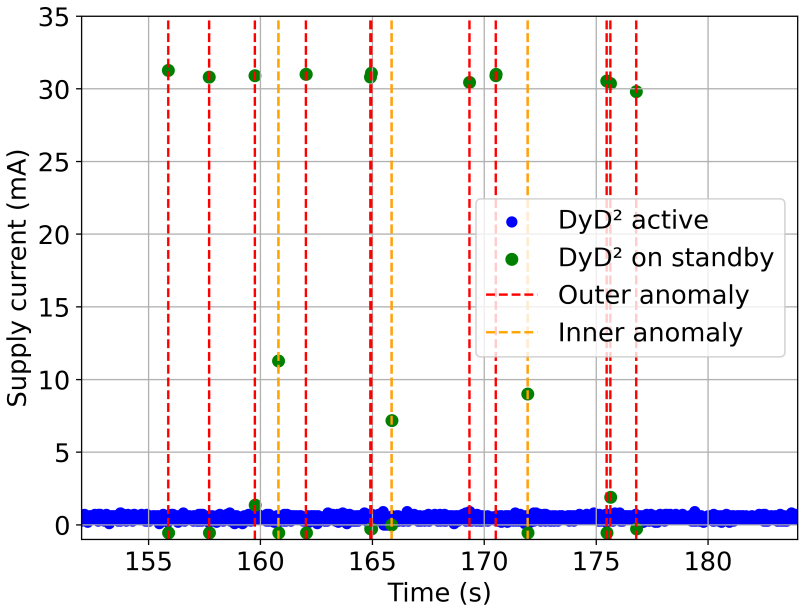


Figure 8.3: Results of DyD² on heavy ion testing

It is possible to conclude that the performance of DyD² obtained with experimental data are similar to the ones obtained with simulated data. It comforts the results obtained solely with simulated data, and the suitability of DyD² for real case applications. The last step is to demonstrate that DyD² is relevant for real-time applications.

8.2.2 Online laser testing

The last aspect that needs to be validated is the suitability of DyD² for real-time applications. In section 7.1, DyD² is described as an on-board algorithm, able to process anomaly detection on incoming observations in real-time. In consequence, DyD² has to be tested on an embedded setup in order to evaluate its capacity to perform in a real-time and memory-limited environment.

To do so, a laser experiment has been carried out. The setup is the same as described in section 5.1.3. The component used to run DyD² is the SAM3X8E microcontroller featured on the Arduino DUE board. It is referenced as the U5 component in figure 5.6. Note that the Device Under Test (DUT) and the microcontroller responsible for the

detection are two different components.

The goal of this experiment is to prove that DyD² is able to process all incoming observations in real-time, with the limitations of the SAM3X8E microcontroller. Thus, only a qualitative approach is used to analyse the results. In consequence, only a single run is presented in this section. During this run, a total of seven anomalies were recorded due to the laser striking a sensitive node on the DUT. Even though those anomalies are not persistent, they represent anomalies that have to be detected by DyD². The sampling time for this experiment was set at 0.1ms.

DyD² is active during the whole run. However, DyD² does not have the possibility to perform a power reset of the DUT when an anomaly is detected. The protection of the component is performed using a threshold-based protection device, the MAX17613 board. During the run, if an anomaly is detected by DyD², the detection is put on hold until the supply current returns to its normal state. In practice, the supply current is considered in its normal state if the last outer sample is reachable by one of the outer map's micro-cluster.

The run is displayed in figure 8.4. As for the heavy ion test performed in section

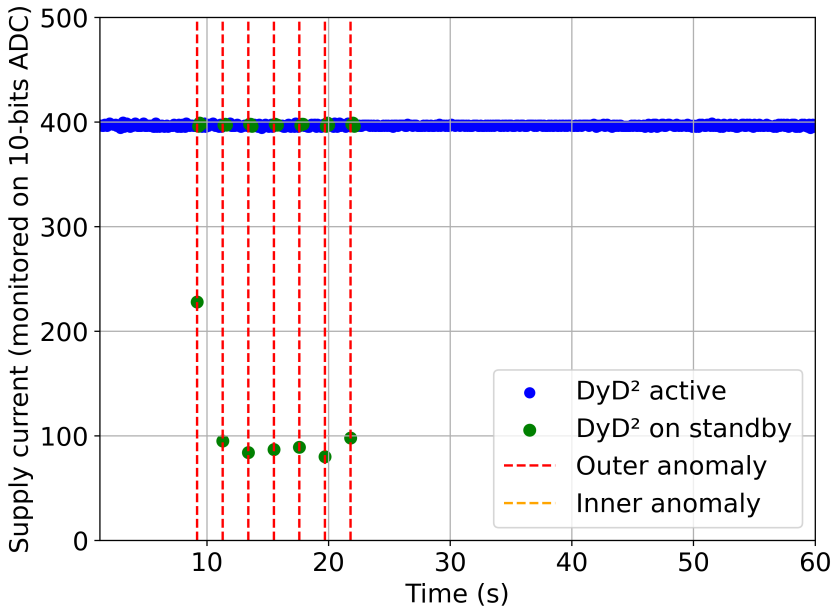


Figure 8.4: Results of DyD² during laser online testing

8.2.1, the monitored supply current is also represented by both blue and green lines. A blue dot indicated that DyD² is active while a green dot indicate that DyD²is on standby and is waiting for the supply current to return to a normal value. The vertical red lines indicate when an outer anomaly is detected. On this run, no inner anomalies were reported by DyD². It is possible to see that DyD² successfully detected all seven

anomalies that happened on the DUT. Also, no other anomalies were reported by DyD^2, meaning that no false positive happened during this run.

In conclusion, DyD^2 was able to run in real-time on a microcontroller limited in memory. It was able to run without fail, and consistently achieved to execute the whole algorithm in less time than the sampling time of 100ms. Therefore, it validates DyD^2 as an on-board anomaly detection algorithm that can be executed on low power components.

8.3 Conclusion

In this chapter, four experiments designed to evaluate the performance of DyD^2 in various situations are reported. First, it has been demonstrated that DyD^2 is on par with the state-of-the-art OCCs algorithms selected for stationary data sets. Indeed, all three criteria were validated by DyD^2 for this experiment. Moreover, an experiment using experimental observations coming from heavy ion testing consolidated these results, as DyD^2 was still able to validate all criteria.

In addition, DyD^2 has been able to adapt to dynamic environment. Indeed, DyD^2 validates all criteria even on data sets including a linear deviation. As a comparison, all selected OCC algorithms failed to validate all criteria for this experiment.

Finally, an experiment conducted on an embedded application demonstrated that DyD^2 is still able to run even in real-time and low-power conditions.

Overall, these results point out that DyD^2 is fitted to perform on-board anomaly detection on dynamic systems. Doing so, it validates DyD^2 as a perfectly functional and efficient anomaly detection method.

Part IV

Conclusion

Chapter 9
Conclusions and perspectives

Contents

9.1 Conclusion

The protection of electronic components is an important topic when designing a space application. With the various effects due to the presence of radiation and high energetic particles, it can be tedious to select the best approach. Moreover, new research emerges regularly to improve the reliability of space electronics during missions. In this manuscript, machine learning methods for anomaly detection were experimented to improve single event effect detection.

Databases gathering observations of normal and faulty behaviour in the supply current of components were created. To do so, an experimental circuit based on the ATMEL SAM3X8E microcontroller was used. Then experimental testing was performed to gather observations of single event effects. By doing so, it was possible to develop a supply current simulator that enables the creation of significant and varied databases. After that, a thorough study of the characteristics of single event effects was reported. Consequently, it was deduced that statistical indicators as well as the frequency spectrum of the supply current are precious information to discriminate single event effects.

Then, the databases were used to evaluate the performance of a selection of anomaly detection methods for single event effects detection. A proof of concept is provided to assert the validity of this approach. Three criteria were presented to compare anomaly detection methods with the baseline detection method used in space applications. Then, the results were divided into three case studies, each of them characterised by the information available during the training phase. Results demonstrated that machine learning-based anomaly detection can be a powerful asset when trying to detect single event effects based on a component supply current.

Finally, a specific algorithm was developed to meet space application requirements. Called Dynamic Double anomaly Detection (DyD^2), this method resolves around the concept of μ-clusters to detect anomalies accurately in time series. DyD^2 is designed

to be functional even on real-time embedded applications, in addition to being able to follow deviations present in dynamic systems. Four experiments were conducted to assess the performance of DyD^2, each focusing on a specific aspect of DyD^2. Results demonstrated that the performance of DyD^2 are on par with the baseline detection method as well as with other selected anomaly detection methods, while being suitable for real-time embedded applications. In conclusion, DyD^2 can be considered as an efficient solution for space applications.

9.2 Perspectives

Some ideas were left unexplored during this project.
First, with promising results of frequency features in the characterisation of single event effects, it would also be interesting to investigate the wavelet function. Indeed, the Fourier transform studied in section 5.4.2 is a transformation primarily focused on periodic signals. However, it is not necessarily the case of the supply current when focusing on low frequency. Using wavelet transformation, it might be possible to uncover decisive characteristics to improve the detection of single event effects.

Another interesting research orientation to explore would be deep learning. Indeed, as stated in section 7.1 of DyD^2, the use of neural networks model was not considered. Two reasons were at the core of this decision. First, because deep models are considered as black-box models, it can be complicated to incorporate them into a space mission for certification reasons. Indeed, the space community prefers to rely on interpretable models when designing protections. The second reason is linked to the calculation power. As of today, space components are still limited, and it is intricate to run complex neural networks on admissible processors. However, with the work of this thesis, machine learning methods might be accepted in the future by the space community, and space components will be powerful enough to run complex models. In that case, embedded deep models are probably the next step for the space industry. Note that some deep models, such as recurrent neural networks for regression, were tried during this project, and showed promising results. Nevertheless, this thesis work should be considered as preliminary work towards the acceptance of machine learning for single event effects detection.

Regarding the DyD^2 algorithm itself, some results of dynamic experiments highlighted that for severe deviations in the data sets, DyD^2 becomes unstable and unable to follow the deviation. It results in a high number of false positives. Nevertheless, a lead is currently being investigated to counter this problem. The properties of the Christoffel function in regard to data sets [125, 126] are considered to detect when a detection map is unable to follow the deviation.
Finally, the last perspective is about the use of the main contribution, DyD^2, on real case space applications. Indeed, many tests have been performed to demonstrate the validity of DyD^2, so only the real test on a satellite is left to do, representing the consecration of this thesis work.

Appendices

Appendix A
Reinforcement Learning

Reinforcement learning is a unique field of machine learning. Here, no database is needed at the beginning of the training. Instead, a learning environment in which an agent can interact to receive *rewards* is used. From there, the agent's ultimate goal is to maximise the reward. Reinforcement learning is a particular and exciting case of machine learning.

Reinforcement learning can be modelled as a *Markov decision process (MDP)*. A MDP is a discrete-time stochastic control process that depends on four factors: a state-space S, an action space A, the probability $P(s'|s,a)$ of reaching a state s' from s using the action a, and the expected immediate reward $R_a(s',s)$.
To take an example, picture an adventurer lost in a 2-dimensional labyrinth. $s \in S$ represents its current location. For each location, the adventurer can choose to go up, down, left, or right. The actions possible for each state are represented by $a \in A$. However, as our adventurer is scared and fallible, he does not always go in the wanted direction. This probability is denoted by $P(s'|s,a)$. Furthermore, as the adventurer is evolving in a Markovian model, the probability of reaching s' from s is only influenced by s, and not from the history of previous states. Finally, a reward is associated with each step taken by the adventurer. The reward can be positive, like getting closer to the exit, or negative, like falling into a deadly trap.

The purpose of reinforcement learning is for an agent to learn through trial and error an optimal policy that maximises the expected reward sum. However, at the beginning, the agent does only know about each possible action for its current state. Reinforcement learning is like playing a game whose rules are unknown to you, and after 15 minutes, you are congratulated with a "You loose" screen. You might get frustrated and give up, whereas an artificial agent will not give up, and will retry as many times as needed to find the best way to win the game.
In reinforcement learning, the developer has to provide a list of rewards that will influence the agent in its environment. It is the key point of reinforcement learning, as it will affect how the agent interprets the rules.

An essential aspect of reinforcement learning is *exploration vs exploitation*. Exploitation consists of an agent using the knowledge uncovered before and starts to optimise its actions to maximise the reward. However, it is possible that the agent gets stuck in a local minimum. To avoid this, the agent has to see as many possibilities as he can. It means that he has to take intentional "bad decisions" to try new options. It is called exploration. The key is to balance exploration so that the agent experiences as much as possible, with exploitation to improve its model.

The most well-known algorithm in reinforcement learning is *Q-learning* [56]. It works by trying to give a score for an action in a particular state. In order to choose the best possible action, the agent uses a function $Q(s, a)$ that represents the expected reward for an action a taken in a state s. At the beginning of training, the function $Q(s, a)$ is initialised randomly. Then after each iteration, it is updated using equation A.1:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[R_a(s', s) + \gamma max_{a'} Q(s', a') - Q(s, a)] \tag{A.1}$$

with α the learning rate, $max_{a'} Q(s', a')$ the maximum reward that can be obtained from the future state s' and $\gamma \in [0, 1]$ the discount factor.

Others example of reinforcement learning algorithms are Deep Q-learning which Google DeepMind first used in 2014 to train an agent to play Atari Breakout and Proximal Policy Optimisation (PPO) [65] developed by OpenAi, which is the currently used algorithm by the game engine Unity. The latter enables the creation of video games that take advantage of reinforcement learning, such as Deep Down by Law Tech Productions, that let the player help a group of neural network-controlled group of adventurers [127].

In this section was given a brief overview of reinforcement learning. The interesting feature of this category lies in the uncertainty of the results. The model can produce unexpected but nonetheless fascinating behaviours.

Appendix B

Schematics

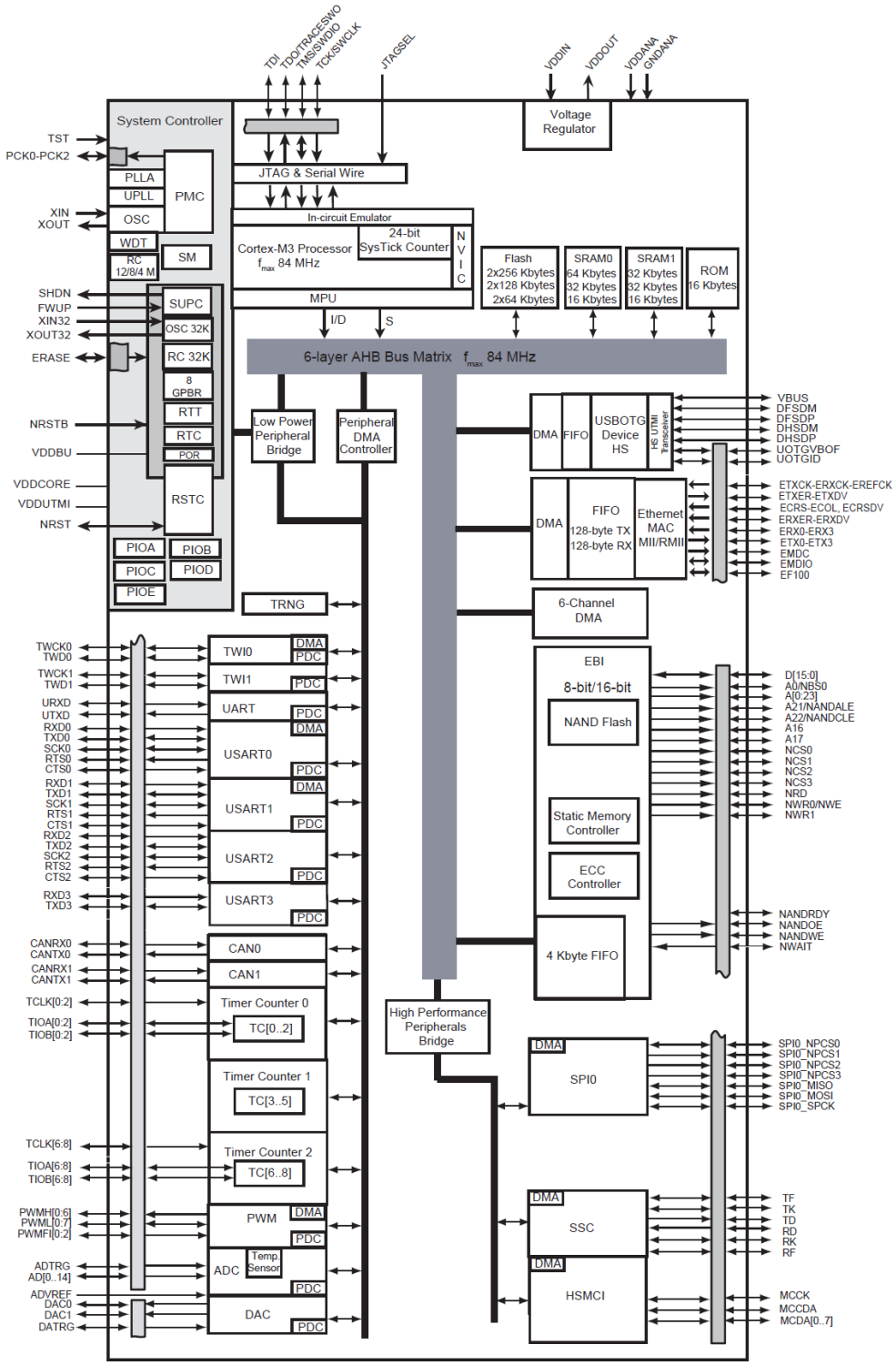


Figure B.1: SAM3X8E bloc diagram

Board	**Name**	Arduino® Due
	SKU	A000062
Microcontroller	AT91SAM3X8E	
USB connector	Micro USB	
Pins	**Built-in LED Pin**	13
	Digital I/O Pins	54
	Analog input pins	12
	Analog output pins	2
	PWM pins	12
Communication	**CAN**	Yes (ext. transceiver needed)
	UART	Yes, 4
	I2C	Yes
	SPI	Yes
Power	**I/O Voltage**	3.3V
	Input voltage (nominal)	7-12V
	DC Current per I/O pin (group 1)	9 mA
	DC Current per I/O pin (group 2)	3 mA
	Power Supply Connector	Barrel Plug
	Total DC Output Current on all I/O lines	130 mA
Clock speed	**Processor**	AT91SAM3X8E 84 MHz
Memory	**AT91SAM3X8E**	96KB SRAM, 512KB flash
Dimensions	**Weight**	36 g
	Width	53.3 mm
	Length	101.5 mm

Figure B.2: Arduino DUE specifications

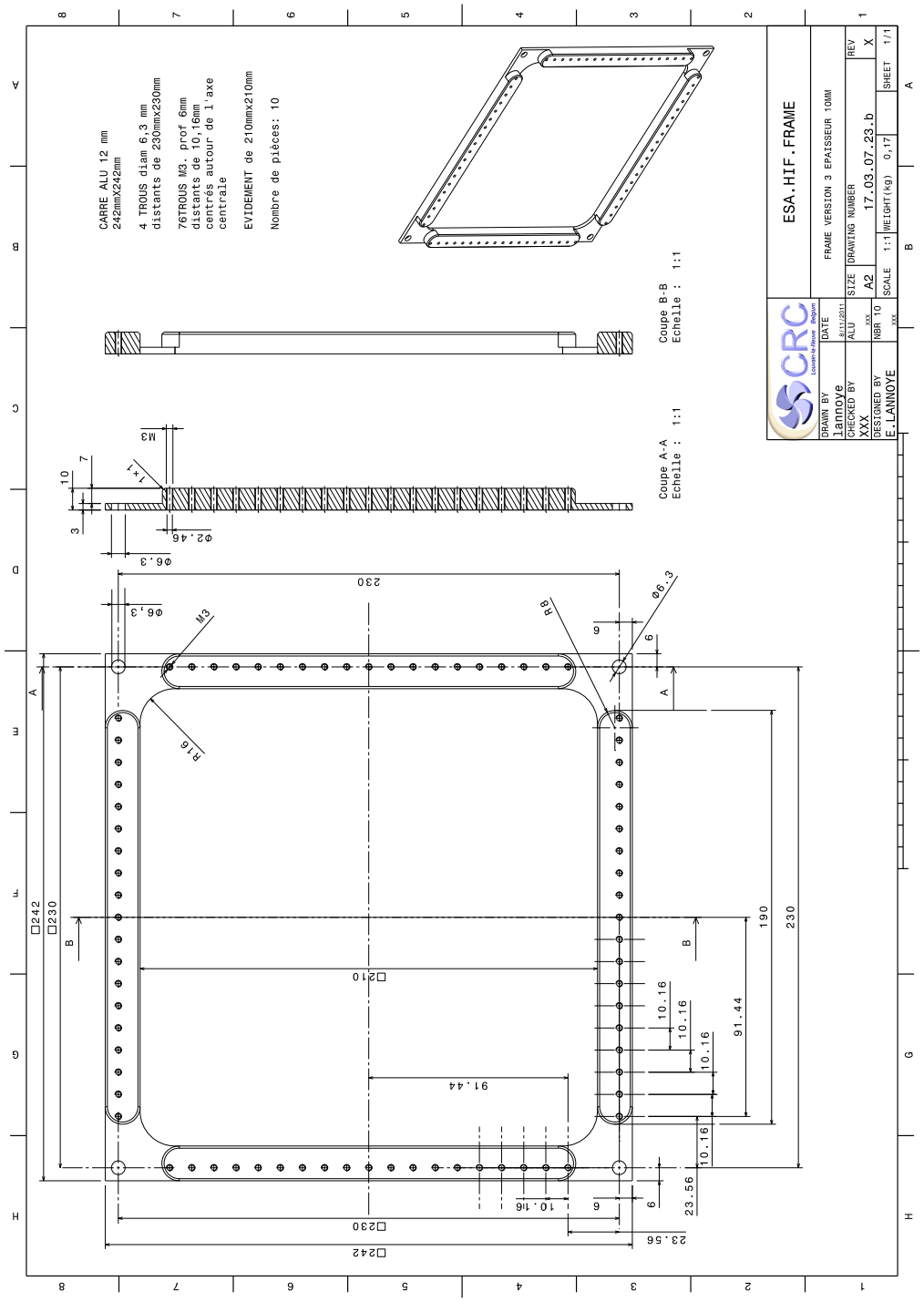


Figure B.3: Cyclotron schematic frame

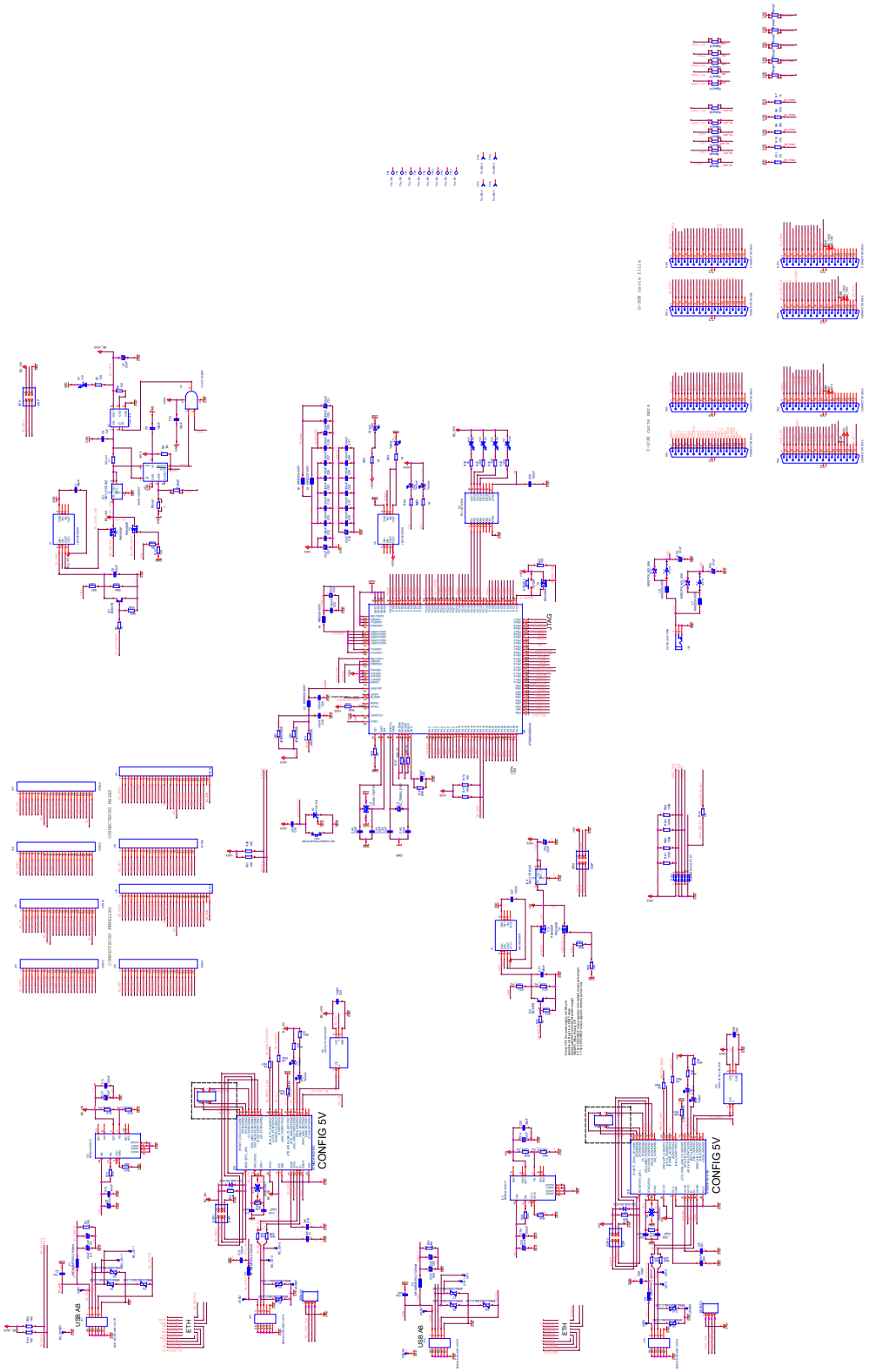


Figure B.4: DIAG-RAD mother board schematic

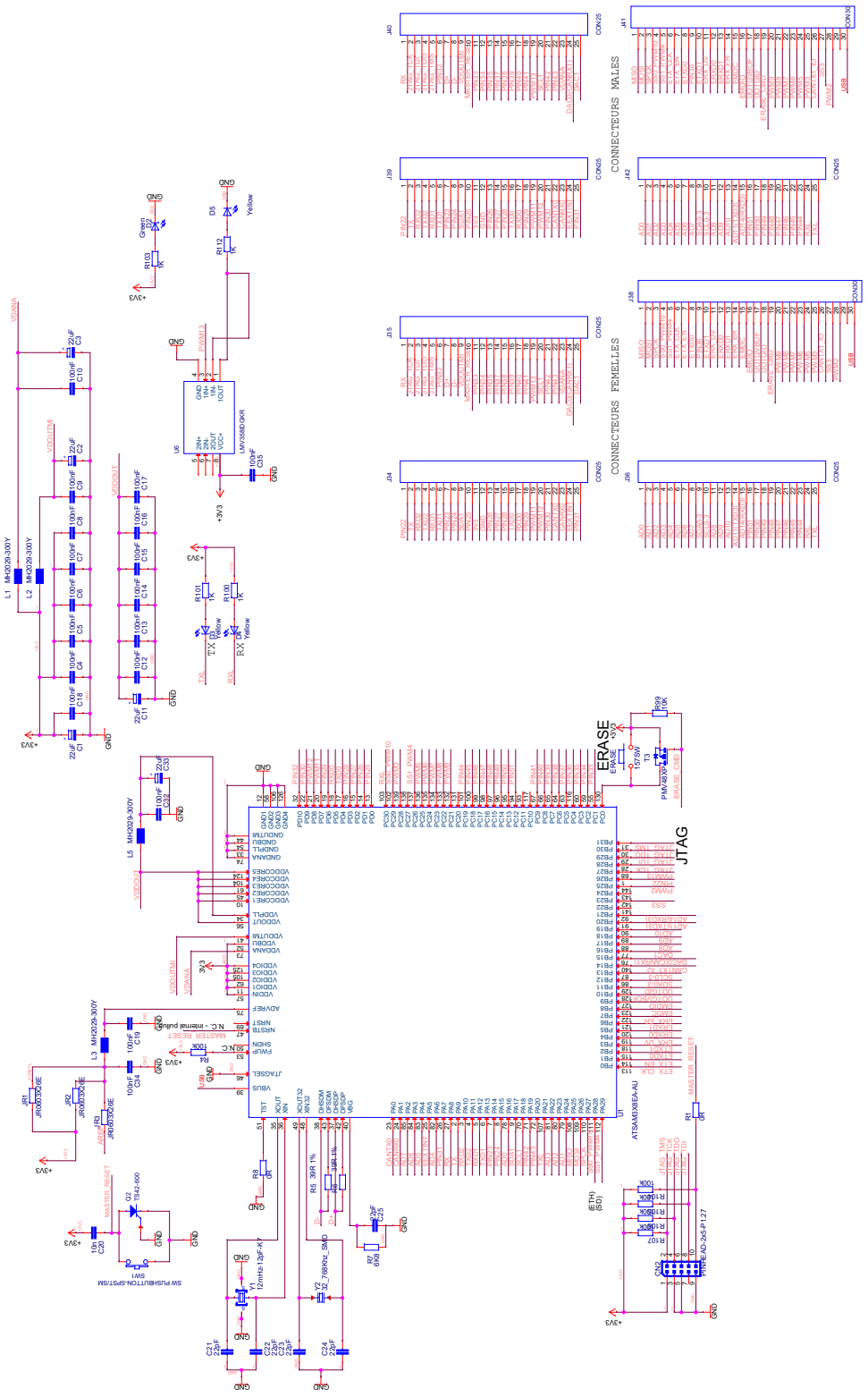


Figure B.5: DIAG-RAD daughter board schematic

Appendix C
Detailed results

Table C.1: Supervised algorithms confusion matrix

		TP	FP	FN	TN	Time
K-NN	$Test_{simD}$	17.33%	2.74%	0.00%	79.94%	0.235s
	$Test_{simND}$	16.91%	5.87%	0.18%	77.04%	0.239s
Naive Bayès	$Test_{simD}$	17.40%	2.75%	0.00%	80.35%	0.108s
	$Test_{simND}$	17.03%	5.54%	0.07%	77.37%	0.260s
Decision tree	$Test_{simD}$	17.33%	3.14%	0.00%	79.53%	0.072s
	$Test_{simND}$	16.65%	8.98%	0.44%	73.93%	0.058s
Random forest	$Test_{simD}$	17.33%	2.72%	0.00%	79.95%	2.885s
	$Test_{simND}$	16.81%	5.31%	0.28%	77.61%	2.912s
SVM	$Test_{simD}$	17.33%	3.50%	0.00%	79.17%	0.104s
	$Test_{simND}$	16.67%	3.96%	0.41%	78.96%	0.177s

Table C.2: Classification boosted by clustering algorithms confusion matrix

		TP	FP	FN	TN
K-Means	$Test_{simD}$	14.71%	5.71%	0.00%	79.58%
	$Test_{simND}$	17.93%	5.87%	0.18%	77.04%
Hierarchical Clustering	$Test_{simD}$	14.71%	5.99%	0.00%	79.30%
	$Test_{simND}$	17.52%	2.80%	0.50%	79.18%
DBSCAN	$Test_{simD}$	14.71%	6.11%	0.00%	79.18%
	$Test_{simND}$	0.00%	0.00%	17.42%	82.58%
DyClee	$Test_{simD}$	14.71%	4.72%	0.00%	80.56%
	$Test_{simND}$	10.95%	1.05%	5.97%	82.02%

Table C.3: One-class classification algorithms confusion matrix

		TP	FP	FN	TN	Time
Elliptic Envelope	$Test_{simD}$	17.33%	0.79%	0.00%	81.88%	0.192s
	$Test_{simND}$	3.66%	0.94%	13.11%	82.30%	0.185s
Isolation Forest	$Test_{simD}$	16.29%	0.90%	1.04%	81.77%	9.691
	$Test_{simND}$	7.55%	0.71%	9.29%	82.45%	12.256s
LOF	$Test_{simD}$	17.33%	7.08%	0.00%	75.60%	0.447s
	$Test_{simND}$	12.11%	6.24%	4.90%	76.76%	0.414s
OC-SVM	$Test_{simD}$	17.33%	3.97%	0.00%	78.71%	0.094s
	$Test_{simND}$	14.07%	2.94%	2.94%	80.04%	0.077s
Auto-encoders	$Test_{simD}$	17.33%	18.30%	0.00%	64.37%	0.174s
	$Test_{simND}$	11.58%	13.36%	5.00%	70.06%	0.097s

Table C.4: DyD^2 and OCC algorithms confusion matrix

		TP	FP	FN	TN	Time
DyD^2	$Test_{simD}$	16.80%	5.02%	0.00%	78.18%	0.034s
	$Test_{simND}$	14.57%	4.46%	1.99%	78.98%	0.036s
	$Test_{simDevD}$	9.06%	9.04%	0.00%	81.91%	0.304s
	$Test_{simDevND}$	9.46%	9.73%	0.10%	80.71%	0.257s
	$Test_{Hion}$	0.41%	0.45%	0.00%	99.14%	NA
Elliptic Envelope	$Test_{simDevD}$	9.06%	16.17%	0.00%	74.10%	1.692s
	$Test_{simDevND}$	7.92%	11.87%	1.69%	78.53%	1.789s
Isolation Forest	$Test_{simDevD}$	9.06%	31.90%	0.0%	58.04%	92.657s
	$Test_{simDevND}$	9.58%	31.12%	0.02%	59.28%	94.161s
LOF	$Test_{simDevD}$	9.06%	41.96%	0.00%	48.98%	4.309s
	$Test_{simDevND}$	9.40%	37.13%	0.21%	53.27%	4.263s
OC-SVM	$Test_{simDevD}$	9.06%	24.18%	0.00%	66.76%	1.048s
	$Test_{simDevND}$	9.28%	23.20%	0.33%	67.20%	1.045s
Auto-encoders	$Test_{simDevD}$	9.06%	38.61%	0.00%	52.33%	0.342s
	$Test_{simDevND}$	8.92%	36.03%	0.63%	54.40%	0.344s

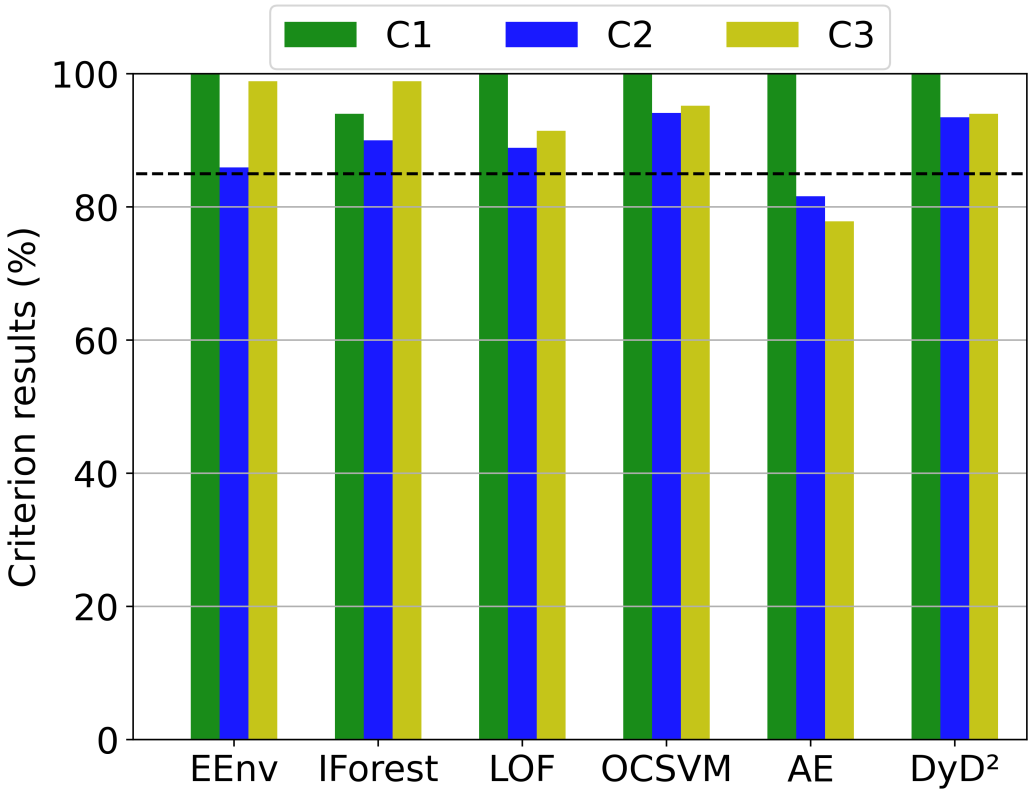


Figure C.1: State of art comparison with DyD^2 on simulated sets

Bibliography

[1] E. Mollick. Establishing moore's law. *IEEE Annals of the History of Computing*, 28(3):62–75, July 2006. ISSN 1934-1547. doi: 10.1109/MAHC.2006.45.

[2] Stephen A. Jacklin. Small-satellite mission failure rates. *NASA technical reports*, NASA/TM-2018-220034, 2019.

[3] Robert Ecoffet. In-orbit anomalies due to radiation. lessons learnt, 03 2011. URL `https://escies.org/download/webDocumentFile?id=49204`.

[4] J. A. VAN ALLEN, G. H. LUDWIG, E. C. RAY, and C. E. McILWAIN. Observation of high intensity radiation by satellites 1958 alpha and gamma. *Journal of Jet Propulsion*, 28(9):588–592, 1958. doi: 10.2514/8.7396. URL `https://doi.org/10.2514/8.7396`.

[5] L. Fletcher, B. R. Dennis, H. S. Hudson, S. Krucker, K. Phillips, A. Veronig, M. Battaglia, L. Bone, A. Caspi, Q. Chen, and et al. An observational overview of solar flares. *Space Science Reviews*, 159(1-4):19–106, Aug 2011. ISSN 1572-9672. doi: 10.1007/s11214-010-9701-8. URL `http://dx.doi.org/10.1007/s11214-010-9701-8`.

[6] Axel Wittmann and Z. T. Xu. A catalogue of sunspot observations from 165 bc to ad 1684. *Astronomy & Astrophysics Supplement Series*, 70:83–94, 1987.

[7] David H Hathaway. The solar cycle. *Living reviews in solar physics*, 12(1):4, 2015.

[8] SILSO World Data Center. The international sunspot number. *International Sunspot Number Monthly Bulletin and online catalogue*, 1749-2022. URL `https://www.sidc.be/silso/DATA/SN_m_tot_V2.0.txt`.

[9] Victor F. Hess, 11 1936. URL `https://www.nobelprize.org/prizes/physics/1936/hess/facts/`.

[10] University of Delaware. Cosmic rays on spaceship earth. *Space Science Symposium*, 2000. URL `https://neutronm.bartol.udel.edu/catch/cr2.html`.

[11] Safework NSW. Ionising and non-ionising radiation. URL `https://www.safework.nsw.gov.au/hazards-a-z/ionising-and-non-ionising-radiation`. Visited in 2022.

[12] Rémi Gaillard. *Single Event Effects: Mechanisms and Classification*, pages 27–54. Springer US, Boston, MA, 2011. ISBN 978-1-4419-6993-4. doi: 10.1007/978-1-4419-6993-4_2. URL `https://doi.org/10.1007/978-1-4419-6993-4_2`.

[13] J. T. Wallmark and S. M. Marcus. Minimum size and maximum packing density of nonredundant semiconductor devices. *Proceedings of the IRE*, 50(3):286–298, 1962. doi: 10.1109/JRPROC.1962.288321.

[14] D. Binder, E. C. Smith, and A. B. Holman. Satellite anomalies from galactic cosmic rays. *IEEE Transactions on Nuclear Science*, 22(6):2675–2680, 1975. doi: 10.1109/TNS.1975.4328188.

[15] T.C. May and M.H. Woods. Alpha-particle-induced soft errors in dynamic memories. *IEEE Transactions on Electron Devices*, 26(1):2–9, 1979. doi: 10.1109/T-ED.1979.19370.

[16] J. C. Pickel and J. T. Blandford. Cosmic ray induced errors in mos memory cells. *IEEE Transactions on Nuclear Science*, 25(6):1166–1171, 1978. doi: 10.1109/TNS.1978.4329508.

[17] C. S. Guenzer, E. A. Wolicki, and R. G. Allas. Single event upset of dynamic rams by neutrons and protons. *IEEE Transactions on Nuclear Science*, 26(6):5048–5052, 1979. doi: 10.1109/TNS.1979.4330270.

[18] E. Petersen, R. Koga, Munir Shoga, J. Pickel, and W. Price. The single event revolution. *IEEE Transactions on Nuclear Science*, 60:1824–1835, 06 2013. doi: 10.1109/TNS.2013.2248065.

[19] P.E. Dodd and L.W. Massengill. Basic mechanisms and modeling of single-event upset in digital microelectronics. *IEEE Transactions on Nuclear Science*, 50(3): 583–602, 2003. doi: 10.1109/TNS.2003.813129.

[20] Ygor Aguiar, Alexandra Zimpeck, and Cristina Meinhardt. Reliability evaluation of combinational circuits from a standard cell library. *South Symposium on Microelectronics (SIM)*, 05 2016.

[21] R. Koga, S.H. Penzin, K.B. Crawford, and W.R. Crain. Single event functional interrupt (sefi) sensitivity in microcircuits. In *RADECS 97. Fourth European Conference on Radiation and its Effects on Components and Systems (Cat. No.97TH8294)*, pages 311–318, 1997. doi: 10.1109/RADECS.1997.698915.

[22] F.W. Sexton. Destructive single-event effects in semiconductor devices and ics. *IEEE Transactions on Nuclear Science*, 50:603 – 621, 07 2003. doi: 10.1109/TNS.2003.813137.

[23] Ahmad Al Youssef, Laurent Artola, S. Ducret, and Geoffroy Hubert. Compact modeling of single event latchup of integrated cmos circuit. *IEEE Transactions on Nuclear Science*, PP:1–1, 06 2019. doi: 10.1109/TNS.2019.2920629.

[24] Jonas Birkeland Carlsen. *Design and Validation of Two Single Event Latch-up Protection Solutions. Comparing a New Single Event Latch-up Test Circuit with the IDEAS IDE3466 Single Event Latch-up Detection Module.* PhD thesis, University of Oslo, 2018.

[25] Jean-Marie Lauenstein. *Single-Event Gate Rupture in Power MOSFETs: A New Radiation Hardness Assurance Approach.* PhD thesis, University of Maryland, College Park, 2013.

[26] Maxime Mauguet. *Etude de la génération d'événements singuliers par excitation laser impulsionnel dans des composants silicium utilisés en environnement radiatif.* PhD thesis, INSA Toulouse, 2019. URL `http://www.theses.fr/2019ISAT0012`. 2019ISAT0012.

[27] Aurore Luu. *Méthodologie de prédiction des effets destructifs dus à l'environnement radiatif naturel sur les MOSFETs et IGBTs de puissance.* Theses, Université Paul Sabatier - Toulouse III, November 2009. URL `https://tel.archives-ouvertes.fr/tel-00512340`.

[28] S. Buchner, D. McMorrow, J. Melinger, and A.B. Camdbell. Laboratory tests for single-event effects. *IEEE Transactions on Nuclear Science*, 43(2):678–686, 1996. doi: 10.1109/23.490911.

[29] D. H. Habing. The use of lasers to simulate radiation-induced transients in semiconductor devices and circuits. *IEEE Transactions on Nuclear Science*, 12(5): 91–100, 1965. doi: 10.1109/TNS.1965.4323904.

[30] M. Mauguet, D. Lagarde, F. Widmer, N. Chatry, X. Marie, E. Lorfevre, F. Bezerra, R. Marec, and P. Calvel. Single events induced by heavy ions and laser pulses in silicon schottky diodes. *IEEE Transactions on Nuclear Science*, 65(8):1768–1775, 2018. doi: 10.1109/TNS.2018.2813096.

[31] Emeric Faraud, Vincent Pouget, Kai Shao, C. Larue, Frédéric Darracq, Dean Lewis, Anne Samaras, Francoise Bezerra, E. Lorfèvre, and R. Ecoffet. Investigation on the sel sensitive depth of an sram using linear and two-photon absorption laser testing. *IEEE Transactions on Nuclear Science*, 58:2637–2643, 2011.

[32] Alexander I. Chumakov, Alexander A. Pechenkin, Dmitry V. Savchenkov, Andrey V. Yanenko, Leonid N. Kessarinskiy, Pavel V. Nekrasov, Armen V. Sogoyan, Alexander I. Tararaksin, Alexey L. Vasil'ev, Vasily S. Anashin, and Pavel A. Chubunov. Compendium of see comparative results under ion and laser irradiation. In *2013 14th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–4, 2013. doi: 10.1109/RADECS.2013.6937390.

[33] D. Lewis, V. Pouget, F. Beaudoin, P. Perdu, H. Lapuyade, P. Fouillat, and A. Touboul. Backside laser testing of ics for set sensitivity evaluation. *IEEE Transactions on Nuclear Science*, 48(6):2193–2201, 2001. doi: 10.1109/23.983195.

[34] Giovanni Palmerini and Francesco Pizzirani. Design of the radiation shielding for a microsatellite. *Acta Astronautica*, 50:159–166, 02 2002. doi: 10.1016/S0094-5765(01)00151-5.

[35] Ian T. Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016. doi: 10.1098/rsta.2015.0202. URL `https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2015.0202`.

[36] Shehroz S. Khan and Michael G. Madden. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(3):345–374, Jan 2014. ISSN 1469-8005. doi: 10.1017/s026988891300043x. URL `http://dx.doi.org/10.1017/S026988891300043X`.

[37] Sylvain Fuertes, Gilles Picart, Jean-Yves Tourneret, Lotfi Chaari, André Ferrari, and Cédric Richard. Improving Spacecraft Health Monitoring with Automatic Anomaly Detection Techniques. In *14th International Conference on Space Operations (SpaceOps 2016)*, page pp. 1, Daejeon, South Korea, May 2016. URL `https://hal.archives-ouvertes.fr/hal-01490731`.

[38] Jize JIANG Joseph Sylvester CHANG, Wei SHU. Electronic circuit for single-event lacth-up detection and protection, 04 2017.

[39] Debra Werner. Airbus ventures adds zero-error systems to space portfolio. *SpaceNews*, 2020.

[40] Roberto Cibils. Method for updating the reference threshold of at least one operational parameter, protection unit for the mitigation of a single event latchup (sel) in an electronic device using the reference threshold and arrangement for the mitigation of a single event latchup (sel) in an array., 05 2021.

[41] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson, 2020. ISBN 9780134610993. URL `http://aima.cs.berkeley.edu/`.

[42] Avneet Pannu and M. tech Student. Survey on expert system and its research areas. *International Journal of Engineering and Innovative Technology (IJEIT)*, 4: 104–108, 2015. ISSN 2277-3754.

[43] T Mitchell, B Buchanan, G DeJong, T Dietterich, P Rosenbloom, and A Waibel. Machine learning. *Annual Review of Computer Science*, 4(1):417–433, 1990. doi: 10.1146/annurev.cs.04.060190.002221. URL `https://doi.org/10.1146/annurev.cs.04.060190.002221`.

[44] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), July 2009. ISSN 0360-0300. doi: 10.1145/1541880.1541882.

[45] Frederic B. Fitch. Warren s. mcculloch and walter pitts. a logical calculus of the ideas immanent in nervous activity. bulletin of mathematical biophysics, vol. 5 (1943), pp. 115–133. *Journal of Symbolic Logic*, 9(2):49–50, 1944. doi: 10.2307/2268029.

[46] A. M. TURING. I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, LIX(236):433–460, 10 1950. ISSN 0026-4423. doi: 10.1093/mind/LIX.236.433. URL `https://doi.org/10.1093/mind/LIX.236.433`.

[47] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959. doi: 10.1147/rd.33.0210.

[48] F ROSENBLATT. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386—408, November 1958. ISSN 0033-295X. doi: 10.1037/h0042519. URL `https://doi.org/10.1037/h0042519`.

[49] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967. doi: 10.1109/TIT.1967.1053964.

[50] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.

[51] Bruce Buchanan, E.A. Feigenbaum, and J. Lederberg. Heuristic dendral: A program for generating explanatory hypotheses in organic chemistry. *Machine Intelligence*, 4, 02 1968.

[52] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[53] G. E. Hinton and T. Sejnowski. Learning and relearning in boltzmann machines. In *Parallel distributed processing: Explorations in the microstructure of cognition*, pages 282–317–. MIT Press, Cambridge, MA, 1986.

[54] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern.*, 59(4–5):291–294, sep 1988. ISSN 0340-1200. doi: 10.1007/BF00332918. URL `https://doi.org/10.1007/BF00332918`.

[55] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.

[56] Christopher Watkins. *Learning From Delayed Rewards*. PhD thesis, Cambridge, 01 1989.

[57] J. Hendler. Avoiding another ai winter. *IEEE Intelligent Systems*, 23(02):2–4, mar 2008. ISSN 1941-1294. doi: 10.1109/MIS.2008.20.

[58] Corinna Cortes and Vladimir Naumovich Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[59] R. A. FISHER. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936. doi: https://doi.org/10.1111/j.1469-1809.1936.tb02137.x. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x`.

[60] Tin Kam Ho. Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95, page 278, USA, 1995. IEEE Computer Society. ISBN 0818671289.

[61] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[62] Quentin Ricard. *Détection autonome de trafic malveillant dans les réseaux véhiculaires*. Theses, Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier), Sep 2020. URL `https://hal.laas.fr/tel-02966530`.

[63] Frederic Chatrie. *Dosimétrie in-vivo et contrôle qualité en radiothérapie externe par réseaux de neurones*. PhD thesis, Université Toulouse 3 Paul Sabatier, 2021.

[64] Edouard Villain. *Utilisation de l'intelligence arti(cielle pour l'aide au diagnostic des patients atteints de pathologies neuro dégénératives*. PhD thesis, Université Toulouse 3 Paul Sabatier, 2021.

[65] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL `https://arxiv.org/abs/1707.06347`.

[66] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, New York, NY, USA, 2016.

[67] Benoit Baudry and Martin Monperrus. Exhaustive survey of rickrolling in academic literature. *Proceedings of SIGBOVIK*, 2022. doi: 10.48550/ARXIV.2204.06826. `https://www.youtube.com/watch?v=dQw4w9WgXcQ`.

[68] Dastan Maulud and Adnan M. Abdulazeez. A review on linear regression comprehensive in machine learning. *Journal of Applied Science and Technology Trends*, 1(4):140–147, Dec. 2020. doi: 10.38094/jastt1457. URL `https://jastt.org/index.php/jasttpath/article/view/57`.

[69] Mark Rafferty, Paul Brogan, John Hastings, D.M. Laverty, Xueqin Liu, and Rafiullah Khan. Local anomaly detection by application of regression analysis on pmu data. *IEEE Power and Energy Society General Meeting (PESGM)*, pages 1–5, 08 2018. doi: 10.1109/PESGM.2018.8586320.

[70] Xiufeng Liu and Per Sieverts Nielsen. Regression-based online anomaly detection for smart grid data, 2016. URL `https://arxiv.org/abs/1606.05781`.

[71] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992. doi: 10.1080/00031305.1992.10475879.

[72] J. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[73] Michael Nielsen. Neural networks and deep learning, 2015. URL `http://neuralnetworksanddeeplearning.com/`.

[74] Xin Jin and Jiawei Han. *K-Means Clustering*, pages 563–564. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_425. URL `https://doi.org/10.1007/978-0-387-30164-8_425`.

[75] J MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297, 1967.

[76] Fionn Murtagh and Pedro Contreras. Methods of hierarchical clustering. *Computing Research Repository - CORR*, 04 2011. doi: 10.1007/978-3-642-04898-2_288.

[77] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[78] Nathalie Barbosa Roa, Louise Travé-Massuyès, and Victor Hugo Grisales. DyClee: Dynamic clustering for tracking evolving environments. *Pattern Recognition*, 94: 162–186, October 2019. doi: 10.1016/j.patcog.2019.05.024.

[79] Martin Bauw, Santiago Velasco-Forero, Jesus Angulo, Claude Adnet, and Olivier Airiau. From unsupervised to semi-supervised anomaly detection methods for hrrp targets, 2021. URL `https://arxiv.org/abs/2106.11168`.

[80] Pramuditha Perera, Poojan Oza, and Vishal M. Patel. One-class classification: A survey. *CoRR*, abs/2101.03064, 2021. URL `https://arxiv.org/abs/2101.03064`.

[81] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008. doi: 10.1109/ICDM.2008.17.

[82] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, may 2000. ISSN 0163-5808. doi: 10.1145/335191.335388. URL `https://doi.org/10.1145/335191.335388`.

[83] Jinghui Chen, Saket Sathe, Charu Aggarwal, and Deepak Turaga. Outlier detection with autoencoder ensembles. *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 90–98, 06 2017. doi: 10.1137/1.9781611974973.11.

[84] Laura M. Ferrari, Guy Abi Hanna, Paolo Volpe, Esma Ismailova, François Bremond, and Maria A. Zuluaga. One-class autoencoder approach for optimal electrode set-up identification in wearable eeg event monitoring, 2021. URL `https://arxiv.org/abs/2104.04546`.

[85] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey, 2006.

[86] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In Jan Van den Bussche and Victor Vianu, editors, *Database Theory — ICDT 2001*, pages 420–434, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44503-6.

[87] Pedro Domingos. A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87, oct 2012. ISSN 0001-0782. doi: 10.1145/2347736.2347755. URL `https://doi.org/10.1145/2347736.2347755`.

[88] Evgeny M. Mirkes, Jeza Allohibi, and Alexander Gorban. Fractional norms and quasinorms do not help to overcome the curse of dimensionality. *Entropy*, 22(10), 2020. ISSN 1099-4300. doi: 10.3390/e22101105. URL `https://www.mdpi.com/1099-4300/22/10/1105`.

[89] J. Budroweit N. Aksteiner. Total ionizing dose effects on current sense amplifiers. *Radiation Effects on Components and Systems RADECS*, 2021.

[90] Roberta Pilia, Remi Espinasse, Christophe Poulet, Francoise Bezerra, Laurene Gillot, Benjamin Treuillard, and Simon Dumortier. SEE Radiation Analysis And Mitigation on SAM3X8ERT Microcontroller, 2021.

[91] *Model DAQ6510 Data Acquisition and Multimeter System*. Keithley, April 2018. Rev. A.

[92] *DAQ6510 Data Acquisition and Logging, Multimeter System Datasheet*. Keithley, 2018.

[93] Françoise Bezerra. Evidence of destructive single event latch-up on various devices using tilu2 test system, 03 2015. URL `https://escies.org/webdocument/showArticle?id=1009`.

[94] Adrien Dorise, Corinne Alonso, Audine Subias, Louise Travé-Massuyès, Leny Baczkowski, and François Vacher. Machine learning as an alternative to thresholding for space radiation high current event detection. In *2021 21th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–7, 2021. doi: 10.1109/RADECS53308.2021.9954582.

[95] Kévin Ducharlet, Louise Travé-Massuyès, Marie-Véronique Le Lann, and Youssef Miloudi. Etude des méthodes de détection d'anomalies non supervisées appliquées aux flux de données. *Rencontres des Jeunes Chercheurs en Intelligence Artificielle*, June 2022. hal-03720505.

[96] Adrien Dorise, Audine Subias, Louise Travé-Massuyès, and Corinne Alonso. Advanced machine learning for the detection of single event effects. In *2022 European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, Venice, Italy, October 2022. URL `https://hal.laas.fr/hal-03789895`.

[97] Jimmy Johansson and Camilla Forsell. Evaluation of parallel coordinates: Overview, categorization and guidelines for future research. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):579–588, 2016. doi: 10.1109/TVCG.2015.2466992.

[98] Salima Omar, Asri Ngadi, and Hamid H. Jebur. Machine Learning Techniques for Anomaly Detection: An Overview. *International Journal of Computer Applications*, 79(2):33–41, October 2013. doi: 10.5120/13715-1478.

[99] Nebrase Elmrabit, Feixiang Zhou, Fengyin Li, and Huiyu Zhou. Evaluation of machine learning algorithms for anomaly detection. In *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, pages 1–8, 2020. doi: 10.1109/CyberSecurity49315.2020.9138871.

[100] RAJ KUMAR and VERMA DR RAJESH. Classification algorithms for data mining: A survey. *INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING AND TECHNOLOGY (IJIET)*, 2012.

[101] Sanjay Yadav and Sanyam Shukla. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pages 78–83, 2016. doi: 10.1109/IACC.2016.25.

[102] Daniel Berrar. Cross-validation. *Encyclopedia of Bioinformatics and Computational Biology*, 2018.

[103] Bradley Efron. *The Jackknife, the Bootstrap and Other Resampling Plans*. Society for Industrial and Applied Mathematics, 1982. doi: 10.1137/1.9781611970319. URL `https://epubs.siam.org/doi/abs/10.1137/1.9781611970319`.

[104] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2017.04.070. URL `https://www.sciencedirect.com/science/article/pii/S0925231217309864`. Online Real-Time Learning Strategies for Data Streams.

[105] Zhang Yang, Nirvana Meratnia, and Paul Havinga. An online outlier detection technique for wireless sensor networks using unsupervised quarter-sphere support vector machine. In *2008 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 151–156, 2008. doi: 10.1109/ISSNIP.2008.4761978.

[106] Juliette Dromard, Gilles Roudière, and Philippe Owezarski. Online and scalable unsupervised network anomaly detection method. *IEEE Transactions on Network and Service Management*, 14(1):34–47, 2017. doi: 10.1109/TNSM.2016.2627340.

[107] Rena Nainggolan, Resianta Perangin-angin, Emma Simarmata, and Astuti Feriani Tarigan. Improved the performance of the k-means cluster using the sum of squared error (SSE) optimized by using the elbow method. *Journal of Physics: Conference Series*, 1361(1):012015, nov 2019. doi: 10.1088/1742-6596/1361/1/012015. URL `https://doi.org/10.1088/1742-6596/1361/1/012015`.

[108] Adrien Dorise. Dyd2_dynamic_double_anomaly_detection, 2022. `https://github.com/Adrien-Dorise/DyD2_Dynamic_Double_Anomaly_Detection`.

[109] Adrien Dorise, Louise Travé-Massuyès, Audine Subias, and Corinne Alonso. Dyd^2: Dynamic Double anomaly Detection. In *IFAC Safeprocess 2022 :11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*, Pafos, Cyprus, June 2022. IFAC. URL `https://hal.laas.fr/hal-03609573`.

[110] Marcello Cannone. *Explainable AI for Clustering Algorithms*. PhD thesis, Politecnico di Torino, 2020. URL `https://webthesis.biblio.polito.it/15868/`.

[111] Christoph Molnar. *Interpretable Machine Learning*. 2019.

[112] Meike Nauta, Jan Trienes, Shreyasi Pathak, Elisa Nguyen, Michelle Peters, Yasmin Schmitt, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable AI. *CoRR*, abs/2201.08164, 2022. URL `https://arxiv.org/abs/2201.08164`.

[113] Alejandro Barredo Arrieta, Natalia Díaz Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *CoRR*, abs/1910.10045, 2019. URL `http://arxiv.org/abs/1910.10045`.

[114] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267, 06 2017. doi: 10.1016/j.artint.2018.07.007.

[115] Bin Yu and Karl Kumbier. Veridical data science. *Proceedings of the National Academy of Sciences*, 117(8):3920–3929, 2020. doi: 10.1073/pnas.1901326117. URL `https://www.pnas.org/doi/abs/10.1073/pnas.1901326117`.

[116] G. Teyssière M. Lavielle. Detection of multiple change-points in multivariate time series. *Lithuanian Mathematical Journal*, 46:287–206, 2006. doi: https://doi.org/10.1007/s10986-006-0028-9.

[117] Ryan Prescott Adams and David J. C. MacKay. Bayesian online changepoint detection, 2007. URL `https://arxiv.org/abs/0710.3742`.

[118] Yoshinobu Kawahara, Takehisa Yairi, and Kazuo Machida. Change-point detection in time-series data based on subspace identification. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 559–564, 2007. doi: 10.1109/ICDM.2007.78.

[119] Xiaofeng Shao and Xianyang Zhang. Testing for change points in time series. *Journal of the American Statistical Association*, 105(491):1228–1240, 2010. doi: 10.1198/jasa.2010.tm10103. URL `https://doi.org/10.1198/jasa.2010.tm10103`.

[120] Dang-Hoan Tran. Automated change detection and reactive clustering in multivariate streaming data. In *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–6, 2019. doi: 10.1109/RIVF.2019.8713738.

[121] D.J. Cook S. Aminikhanghahi. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51:339–367, 2017. doi: https://doi.org/10.1007/s10115-016-0987-z.

[122] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020. ISSN 0165-1684. doi: https://doi.org/10.1016/j.sigpro.2019.107299.

[123] Richard Hyde, Plamen Angelov, and Angus Robert MacKenzie. Fully online clustering of evolving data streams into arbitrarily shaped clusters. *Information Sciences*, 382:96–114, 2017.

[124] Simon Rubinstein-Salzedo. *Big O Notation and Algorithm Efficiency*, pages 75–83. Springer International Publishing, Cham, 2018. ISBN 978-3-319-94818-8. doi: 10.1007/978-3-319-94818-8_8. URL `https://doi.org/10.1007/978-3-319-94818-8_8`.

[125] Kévin Ducharlet, Louise Travé-Massuyès, Jean-Bernard Lasserre, Marie-Véronique Le Lann, and Youssef Miloudi. Leveraging the Christoffel-Darboux Kernel for Online Outlier Detection. working paper or preprint, February 2022. URL `https://hal.laas.fr/hal-03562614`.

[126] Jean-Bernard Lasserre. The Christoffel-Darboux Kernel for Data Analysis. In *23ème congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Villeurbanne - Lyon, France, February 2022. INSA Lyon. URL `https://hal.archives-ouvertes.fr/hal-03595424`.

[127] Adrien Dorise and Ronan Pons. Deep down, 2021. `https://doriens.itch.io/deep-down`.