POLITECNICO DI TORINO

Master's Degree in Mechatronics Engineering

Master's Degree Thesis

Design of a monitoring tool for the health assessment of power converters

Supervisors

Prof. Igor Simone STIEVANO

Prof. Paolo MANFREDI

Candidate

Mattia Rossi

Abstract

Power electronics, in particular power converters, have a pivotal role in multiple expanding fields like: Electric mobility, renewable energy integration, microgrids, and smart grid technologies. Reliability for power electronic converters is a significant concern, since, compared with other electrical modules, they are characterized by a higher failure rate, as reported in different field reviews. The consequences of a converter failure are not only limited to the downtime of the system and potential damage to downstream components. It can also pose a threat to the safety of the system. Those reasons made the lifetime management of power converters an increasingly studied field.

This master's thesis aims to detect the ageing of a power converter by identifying the parameters that are associated with the degradation of a component. This goal is achieved with the implementation of a digital twin of the circuit. The output voltage is simulated and then compared with the measured output voltage of the converter. Minimizing the cost function built between the two characteristics, with the implementation of an optimizer, provides an estimation of the system parameter.

Initially, the method was tested on a simple RLC circuit where a benchmark was made to choose the most suitable optimizer for this application.

Lastly, once the algorithm proved its efficiency on the RLC circuit, its efficacy was tested on a buck converter. The higher complexity of the system introduced other challenges, one of which was to process the output of the simulation in order to align and normalize the data. A qualitative analysis of the estimation of the failure indicator was then performed.

Table of content

List of Table

List of Figures

List of Acronyms

SiC: Silicon carbide

MOSFET: *Metal-Oxide-Semiconductor Field-Effect Transistor*

ESR: *Equivalent series resistor*

RMSE: *Root mean square error*

Lsqnonlin: *Non-linear least square algorithm*

Chapter 1

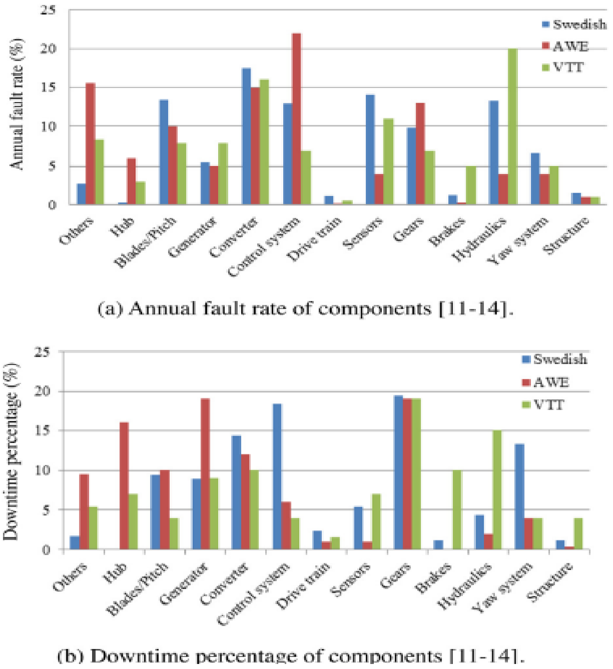
Introduction

1.1 Overview

The switching mode power converters are characterized by relevant features, including high conversion efficiency, low weight, and compact size. Those characteristics made this type of converter a widespread solution for power conversion across different industries, from applications with lower requirements, such as in consumer electronics for domestic appliances or portable electronics, to more critical and demanding systems, such as DC distribution systems or electric motor drives [1].

The widespread adoption of switching-mode power converters in various expanding industry verticals includes electric mobility, renewable energy integration, microgrid, and smart grid technologies has made their reliability a priority [2][3][4]. Reliability for power electronic converters is a significant concern [3] since, compared with other electrical modules, it is characterized by a higher failure rate, as reported in different field reviews.

Two applications where the reliability of power converters is a significant concern are in wind turbines and photovoltaic plants. Wind-power generation systems use power electronic converters to regulate the fluctuating input power and maximize the electrical energy harvested from the wind [5]. As shown in the histogram of *Figure 1*, power converters have a relevant annual failure rate above 15% [6].



(a) Annual fault rate of components [11-14].

(b) Downtime percentage of components [11-14].

Figure 1 Annual fault rate and downtime percentage of components, figure from [6]

Failures of power converters are even more relevant in photovoltaic plants, where the inverters were responsible for 37% of the unscheduled maintenance and 59% of the associated cost [5], as shown in the pie chart in *Figure 2*.

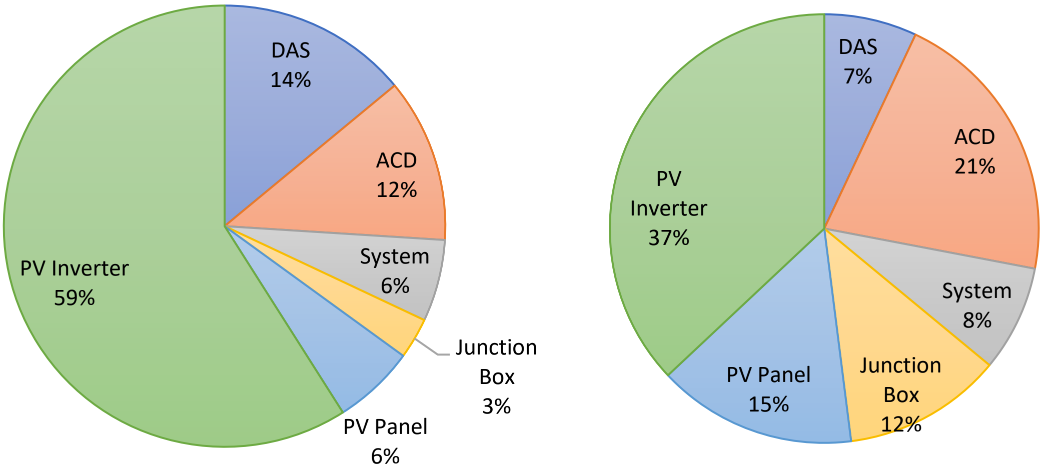


Figure 2 Unscheduled maintenance cost (left) and time (right) in PV application [5]

It has to be noted that the cost associated with an unscheduled interruption due to a fault in a power converter is not restricted to replacing only the faulted converter. Additional costs come from downtime for maintenance but also from downstream components that, in the worst-case scenario, can be damaged by the fault leading to the shutdown of the entire system or starting to work abnormally, impairing the efficiency of the system [7].

For example, in hybrid electric vehicles, a fault in electric propulsion systems impairs fuel economy and lengthens the cost recovery period [3]. It is also important to acknowledge that failures have consequences beyond financial losses since they also pose a threat to the safety of the system [3].

Monitoring the power converter's performance and behavior is a viable path in order to estimate the remaining useful life (RUL) of the system and take preventive measures before anomalies in the equipment turn into system-critical failures [8]. Thus, reducing the unplanned downtime maintenance and have a significant economic improvement.

1.2 Failures classification and management

In order to correctly manage failures in the equipment, it is crucial to comprehend and categorize the mechanism and the mechanisms by which these phenomena occur. The failure mechanism of power converters can be classified as:

- Catastrophic failures
- Ageing failures.

Catastrophic failures often occur without any warning signs. This type of failure happens abruptly with time frames that range between the micro to milliseconds and are more commonly triggered by overstress conditions, such as short circuits or over voltages [2].

Ageing failures occur after several years of operations that lead to a degradation of the device that can be detected with several indicators. The wear-out caused by the long-term application makes the components of the power converter too fragile to withstand nominal electrical and thermal solicitation, and the entire system may collapse [9].

To deal with these two classes of failures, the general guideline for the reliability of power electronic systems split their management as illustrated in *Figure 3*.

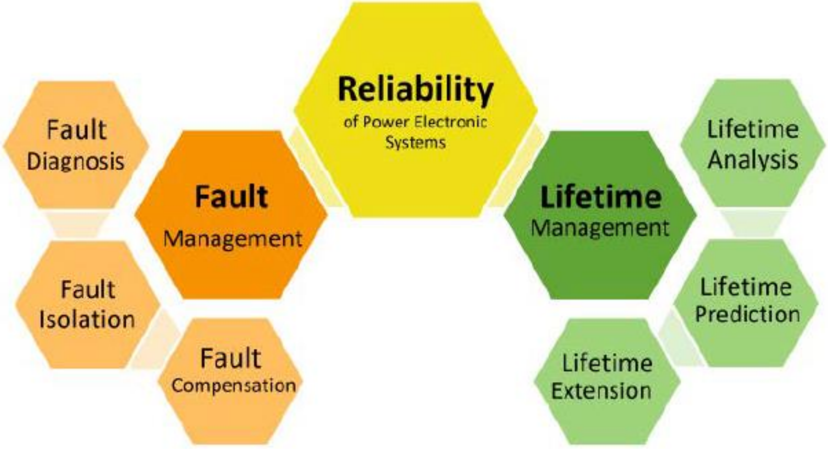


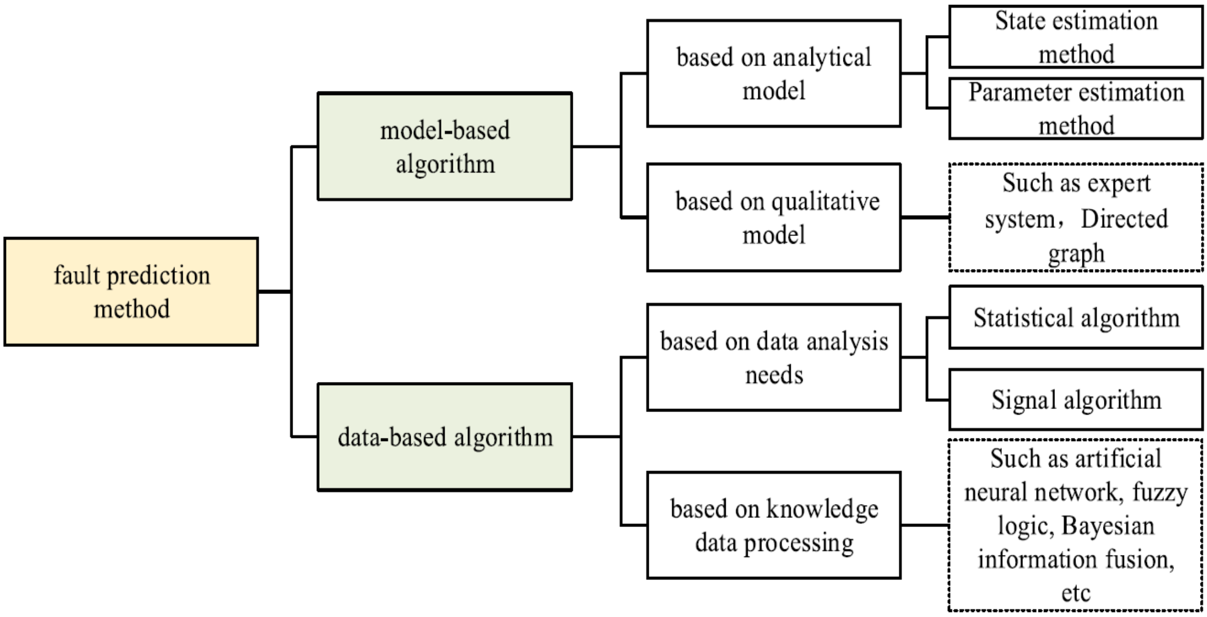
Figure 3 Guideline of the reliability of power electronic system, figure from [3]

Fault management deals with catastrophic failures and its main objectives are: protect the integrity of the system through the implementation of circuit breakers, compensate and to diagnose the fault [3].

On the other hand, lifetime management deals with predicting failures and extending the lifetime of the power converter [3]. Different approaches can be followed in order to predict the remaining useful life of a component and predict a possible failure. Traditionally, those estimations were handbook driven. Nowadays, in the era of the industry 4.0, the required accuracy standard for lifetime expectancy is higher since it allows the transition from preventive maintenance to predictive maintenance [10]. Hence more advanced strategies for fault prediction have been developed.

1.3 Condition monitor strategies

In order to assess the current health state and estimate the future state of the system [10], different strategies were developed. As shown in *Figure 4*, currently, those strategies are divided mainly into two categories: Model-based and data-based.



fault prediction method
model-based algorithm
data-based algorithm
based on analytical model
based on qualitative model
based on data analysis needs
based on knowledge data processing
State estimation method
Parameter estimation method
Such as expert system，Directed graph
Statistical algorithm
Signal algorithm
Such as artificial neural network, fuzzy logic, Bayesian information fusion, etc

Figure 4 Classification of fault prediction methods, figure from [11]

1.3.1 Data-based methods

Data-based methods are focused on data analysis or on data processing. The data analysis approach needs to use historical data to find the corresponding relationship between the statistical characteristics or time domain and frequency domain characteristics of system fault and system state value. After collecting the current system state value, it is analyzed and compared with historical data for fault prediction [11]. On the other hand, the data processing approach utilizes machine learning in order to conduct statistical analysis of historical data with the objective of determining the corresponding relationship between the system fault and the system status. By inputting the current system status, it is possible to predict the system fault information. To implement this strategy, methods like fuzzy logic and neural networks can be applied [11].

1.3.2 Model-based methods

As shown in the classification of *Figure 4,* model-based methods are subdivided into qualitative and analytical models. In order to implement a model-based condition monitoring that uses an analytical model, it is necessary to firstly develop an accurate physical model of the system in exam that is able to take into consideration environmental interference and the wear of the components. The main advantage of this approach is that it can predict the remaining useful life of a component without knowing the system state value. The main drawback is that it cannot account for the influence of the system operation and other modelling factors on the components [11].

To implement a model-based condition monitoring with a qualitative model, it is necessary to consider the fault transmission relationship between nodes when establishing the model in order to predict the impact of the fault of a component on other elements of the system [11].

1.3.3 Digital twin concept

Data-driven techniques for condition monitoring have achieved significant improvement in the last ten years thanks to the substantial advancement in sensor, communication and information technologies and data mining [10]. Nonetheless, those approaches have faced significant challenges due to: Limitation of data availability in different operating condition and the limited ability to provide reliable and interpretable solutions due to the black-box nature of artificial intelligence [10].

A novel approach that has promising capabilities for health monitoring is the digital twin. Firstly, introduced by the U.S. Air Force Research Laboratory as a comprehensive physical and functional description of a component, product, or system [8]. The Digital Twin (DT) is an evolution of the conventional model-based approach that generates a virtual counterpart that mirrors the real system [12]. Compared to data-driven methods, the virtual mirror of the digital twin is able to trace and assess the performance degradation of the single components instead of relying on statistical methods [10]. Thus, this approach is able to take into account the specific operating environment and load usage of the specific machine instead of relying on average prediction based on historical data [10].

In order to apply this concept to a power converter, the first requirement is to establish an accurate mathematical model for the monitored electronic entity [8]. Then, the measurement architecture and a parameter identification method must be chosen. The virtual replica will update its electrical parameters continuously according to the existing measured data from the physical counterpart [13]. Thus, allowing to monitor the parameter evolution of the different components. By analyzing the physic-of-failure, it will be possible to identify which parameters are the failure indicators. Identifying the behavior of those indices and tracking their evolution will become a powerful tool to assess the ageing of the various elements of the converter and predict its remaining useful life.

1.4 Failure mode of critical components

The first step to correctly assess the condition of a power converter is initially to identify the prone-to-failure components [3]. By collecting and analyzing different surveys that studied the failure distribution among power electronic components, it is possible to notice that all of those surveys address the capacitor and the power semiconductor as the most critical element of the system [5],[7],[14],[15]. According to the results of the examined surveys, capacitors and power semiconductors are the cause of at least 50% of the failures in power converters. The pie graph in *Figure 5* depicts one of those surveys' results.

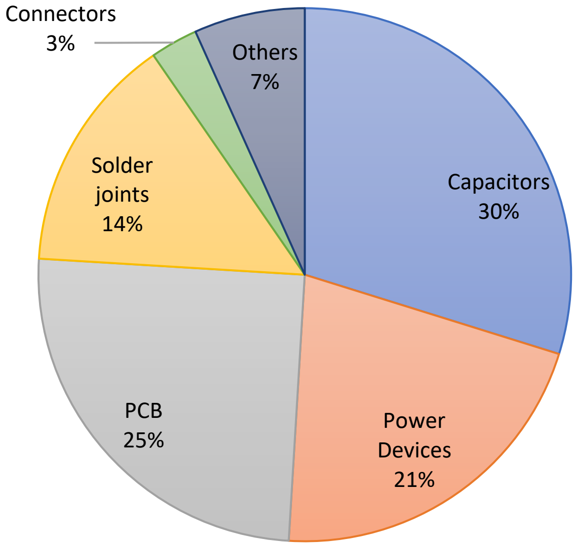


Figure 5 Failure distribution among major components [3]

Since the literature gives a univocal response on which are the prone-to-failure components, it is crucial to study the physics of failure of those components in order to identify which parameters are measurable failure precursors.

1.4.1 Power semiconductor switches

Failure mode in the power semiconductor can occur either at the chip level or at the package level.

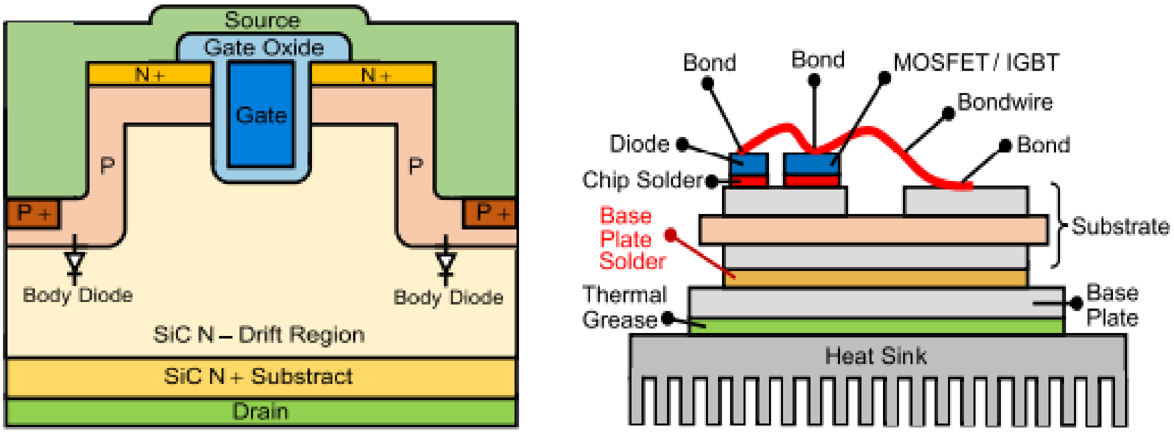
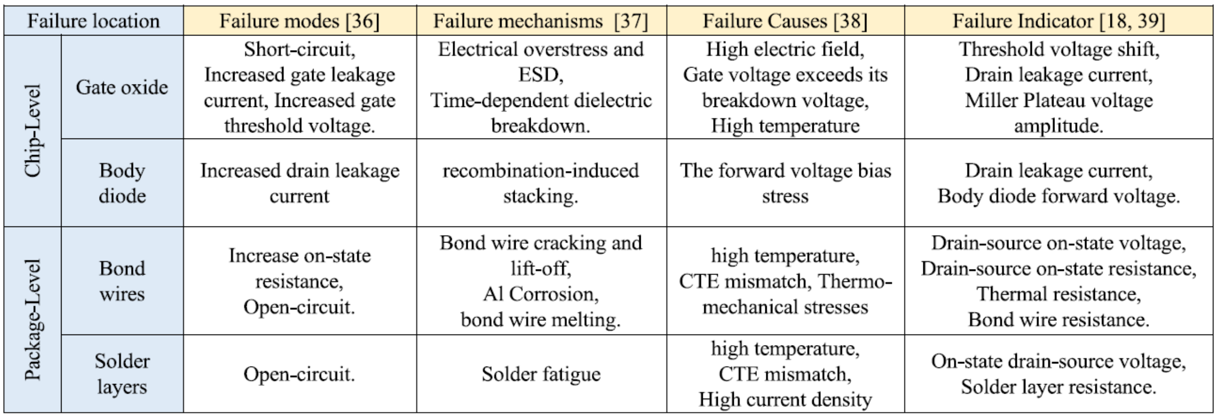


Figure 6 SiC MOSFET chip-level structure (left) package-level structure (right)

The main concern for chip-level failure in SiC MOSFET is gate oxide breakdown. Damage to the gate oxide can occur as the result of overstress that can lead to a catastrophic failure but also due to wear out of the component. Failures caused by the damage accumulated in the gate oxide region of a MOSFET during regular use are referred to as time-dependent breakdown (TDDB) [16]. The damage in the gate oxide can result in excessive leakage current, increased standby power, and decreased response time. Eventually, the damage will cause a MOSFET to short-circuit [16 3].

The other chip-level failure that occurs in SiC MOSFET is the body diode failure, which is more commonly provoked by the recombination-induced stacking fault mechanism. The leading cause of the body diode degradation is the forward voltage bias stress which leads to an increase in forward voltage and drain leakage current [3]. The continuous thermomechanical stresses result in the formation of voids and cracks in the solder layers and reduce the effective area accessible for heat loss reduces, leading to a rise in the module thermal resistance. These further result in an increase in the device junction temperature, which may cause acute localized heating; further possibly leading to catastrophic burnout [17].

The weak spots for package-level failures are bond wires and solder layers. Those elements are vulnerable to thermomechanical fatigue that eventually weakens the metallic contacts. The effect of thermomechanical fatigue can be further exacerbated by humidity and high current density. Humidity can cause a plummet in the metal atom bonding energy. Therefore, the crack growth rate at the tail of the bond wire increases due to atom corrosion [3]. High current density stress leads to acceleration in electromigration-related degradation [18], causing high junction temperatures in bond wires which further leads to the increase of on-state drain-source voltage and resistance [3]. All those types of failure and their characteristics are summarized in the following Table from [3].



Failure location		Failure modes [36]	Failure mechanisms [37]	Failure Causes [38]	Failure Indicator [18, 39]
Chip-Level	Gate oxide	Short-circuit, Increased gate leakage current, Increased gate threshold voltage.	Electrical overstress and ESD, Time-dependent dielectric breakdown.	High electric field, Gate voltage exceeds its breakdown voltage, High temperature	Threshold voltage shift, Drain leakage current, Miller Plateau voltage amplitude.
	Body diode	Increased drain leakage current	recombination-induced stacking.	The forward voltage bias stress	Drain leakage current, Body diode forward voltage.
Package-Level	Bond wires	Increase on-state resistance, Open-circuit.	Bond wire cracking and lift-off, Al Corrosion, bond wire melting.	high temperature, CTE mismatch, Thermo-mechanical stresses	Drain-source on-state voltage, Drain-source on-state resistance, Thermal resistance, Bond wire resistance.
	Solder layers	Open-circuit.	Solder fatigue	high temperature, CTE mismatch, High current density	On-state drain-source voltage, Solder layer resistance.

Table 1 Failure location, modes, mechanism, causes and indicators in MOSFETs. Table from [3]

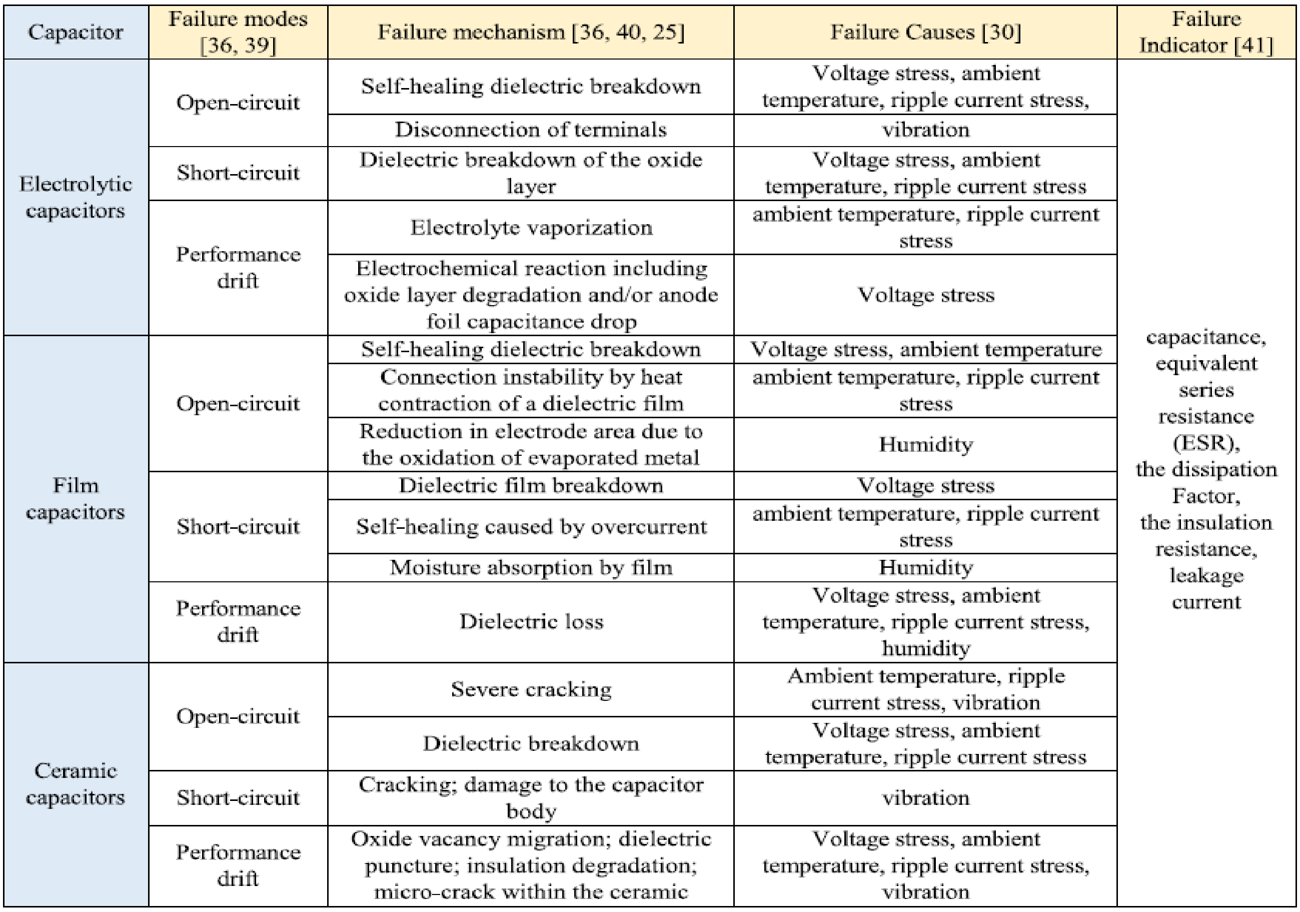
1.4.2 Capacitor

The capacitor is the component that has the most significant impact on the reliability of the power converter. In particular, the electrolytic capacitor, which is the most common in power applications due to its high capacity, is also the one with the higher probability of experiencing wear-out failures due to ageing [19].

The vaporization of the electrolytic fluid is an inevitable consequence of the ageing of the component. This process leads to an increase in the equivalent series resistor (ESR), and since it is inversely proportional to the maximum amount of current the capacitor can handle, the internal temperature of the capacitor begins to increase. As the temperature increases, the likelihood of dielectric breakdown increases. As a result of dielectric breakdown, the capacitor enters a short circuit failure mode [16].

Damage to the dielectric membrane is also caused by external stress like High-temperature thermal shocks and over-voltage. In aluminum electrolytic capacitors, the dielectric membrane damage will initiate a self-healing process that combines oxygen from the electrolyte with the aluminum foils to form Al_2O_3 and hydrogen gas as a waste product. With ageing, this process will increase the rate of vaporisation and degradation of the electrolyte in the capacitor. Under normal operating conditions, the self-healing process is not problematic since the hydrogen produced is negligible. However, once the stress applied to the capacitor becomes relevant, the self-repairing mechanism increases, resulting in more hydrogen gas production and further electrolyte vaporisation that increases the capacitor's internal pressure, ultimately causing the pressure relief vent to open [20].

The processes that lead to the degradation of the electrolytic capacitor are summarized in the following table from [3].



Capacitor	Failure modes [36, 39]	Failure mechanism [36, 40, 25]	Failure Causes [30]	Failure Indicator [41]
Electrolytic capacitors	Open-circuit	Self-healing dielectric breakdown	Voltage stress, ambient temperature, ripple current stress,	capacitance, equivalent series resistance (ESR), the dissipation Factor, the insulation resistance, leakage current
		Disconnection of terminals	vibration	
	Short-circuit	Dielectric breakdown of the oxide layer	Voltage stress, ambient temperature, ripple current stress	
	Performance drift	Electrolyte vaporization	ambient temperature, ripple current stress	
		Electrochemical reaction including oxide layer degradation and/or anode foil capacitance drop	Voltage stress	
Film capacitors	Open-circuit	Self-healing dielectric breakdown	Voltage stress, ambient temperature	
		Connection instability by heat contraction of a dielectric film	ambient temperature, ripple current stress	
		Reduction in electrode area due to the oxidation of evaporated metal	Humidity	
	Short-circuit	Dielectric film breakdown	Voltage stress	
		Self-healing caused by overcurrent	ambient temperature, ripple current stress	
		Moisture absorption by film	Humidity	
	Performance drift	Dielectric loss	Voltage stress, ambient temperature, ripple current stress, humidity	
Ceramic capacitors	Open-circuit	Severe cracking	Ambient temperature, ripple current stress, vibration	
		Dielectric breakdown	Voltage stress, ambient temperature, ripple current stress	
	Short-circuit	Cracking; damage to the capacitor body	vibration	
	Performance drift	Oxide vacancy migration; dielectric puncture; insulation degradation; micro-crack within the ceramic	Voltage stress, ambient temperature, ripple current stress, vibration	

Table 2 Failure location, modes. mechanism, causes and indicators of capacitors. Table from [3]

In all cases, the detection of an incipient failure is made by measuring the ESR since it is a failure indicator that is shared in common between all failure modes and types of capacitors.

1.5 Parameter identification strategies

Parameter identification problems have been studied for a long time, resulting in an extensive number of strategies proposed in the literature to unravel this problem. In the technical bibliography, various approaches are discussed. These include online methods where real-time data are obtained and used immediately to identify the unknown characteristics of the system [21], offline methods, but also methods based on frequency domain or time domain analysis.

In [21], the parameters of a dc-dc converter are identified from the discretized differential equations that describe the dynamic behavior of the converters. This approach allows to obtain both the transfer function coefficient of the controller and the passive component. The drawback of this solution is that it is invasive since it requires measuring the current in the inductor.

A non-traditional approach for parameter estimation is proposed in [22]. In [22], accurate open-loop and closed-loop steady-state models are derived and then used to generate the training data set to obtain the artificial neural network models [22]. The drawback of this solution is that it requires a large dataset of training and test signals under different load conditions [1].

Less exotic solutions, such as recursive least square in [23] and Kalman filter in [24], were also explored. The Kalman filter proved its effectiveness in tuning the controller and improving the performance of the system. However, when mapping the transfer function coefficients, transfer errors could occur for the internal parameters of the converter, and there could also be no feasible solutions when the unknown parameters are more than the known equations [9]. Due to the high nonlinearity of the transfer function coefficients, traditional algorithms, like RLS and Kalman filter, could not succeed in correctly estimating the parameter as demonstrated in [23]. Also, the extended Kalman filter is proposed to solve the nonlinear problem. This approach suffers from instabilities due to the linearization and costly calculation of the Jacobian and Hessian matrices, and its performance deteriorates when the signal model is highly nonlinear [9]. Another critical point for the design of the extended Kalman filter and unscented Kalman filter is determining the covariance matrices of the process noise, the measurement noise, and the initial state vector. This process often requires tedious trial-and-error tuning or self-tuning algorithms [24],[9].

A different type of online identification is proposed in [25], where the identification is based on a dichotomous coordinate descent algorithm and uses an infinite impulse response adaptive filter as the plant model [25].

All these techniques have their respective advantages and disadvantages, which are reviewed in [26]. However, all the mentioned strategies for parameter identification share a common drawback; they are all invasive methods that rely on external excitation signals to effectively estimate the parameter [1]. From the literature review, it also transpired that the researcher focused more on identifying converters through a transfer function, from which it is not always feasible to obtain the values of passive elements and control parameters. In the context of parameter estimation for condition monitoring, the values of the converter's components are ideally identified both in a non-invasive way and directly in order to detect the indices of degradation that were defined by analyzing the physic of failure of the component. A method that satisfies both those requirements is presented in [1]. This method allows the estimate of the values of the internal parameters by minimizing a cost function calculated on the difference between a measured output of the system and the simulated output. The optimization is made by an optimizer algorithm that tunes the parameters of the simulated model.

Chapter 2

Parameter identification

2.1 The identification problem

The strategy implemented to solve the identification problem is a white-box approach that accomplishes the identification of the parameters by minimizing a suitable cost function. Since this is a white-box approach, it is necessary to define the physics of the problem through algebraic or differential equations that describe the system's behavior. The architecture of the power converter and, consequently, the equation that describes the power converter are well known. Still, some considerations must be adopted in order to restrain the computational burden [1].

The parameter identification is made by leveraging the voltage or current ripple that is present at the terminals of any switching mode power converter due to their working principle. The ripple is constituted by transitory behaviors induced in the passive components of the circuit by the periodic switching of the transistors, allowing to obtain, depending on the topology of the converter, higher or lower mean voltages or, in the case of an inverter, to convert continuous current into an approximation of an alternate current.

In switching mode power converters, the duty cycle of the ripple is the controlled variable of the converter that allows the modulation of the output voltage to the desired one. Therefore, when there are no perturbations applied to the load, the converter controller achieves a steady state condition, which means that the duty cycle and, hence, the period of the ripple becomes constant.

The optimization problem is defined in this condition of steady state. The target cost function for the optimization is determined by measuring the ripple in steady-state conditions and then comparing the measured curve with a characteristic that was simulated by using the set parameter chosen for that iteration by the optimizer.

Selecting a suitable cost function for the optimizer is a crucial design choice that could heavily influence the final performance of the parameter estimation. For this reason, multiple tests were performed to assess the performance of different definitions and combinations of cost functions. Once the cost function is defined, the role of the optimizer is to tweak the values of the passive component of the model in order to minimize the cost function and consequently minimize the difference between the observed “target”

characteristics and the simulated one. Matching the output characteristics will provide an estimate of the values of the components that constitute the model.
In papers [9],[1],[13],[8], the objective function was always defined as a sum of squared residuals. In [1], a normalization was applied in order to mix different measurements. In this work, the sum of the square residual $e_t(x)$ is defined as follows:

$$f_{RSS}(x) = \frac{1}{N_{Ttot}} \sum_{t=1}^{T_{tot}} e_t^2 \ x \tag{2.1}$$

Where $\boldsymbol{x}$ is the vector that contains all the parameters that will be estimated and $e_t(x)$ is the difference between the simulated and observed transitory curve at the instant t. A representation of this type of cost function is depicted in *Figure 9.*

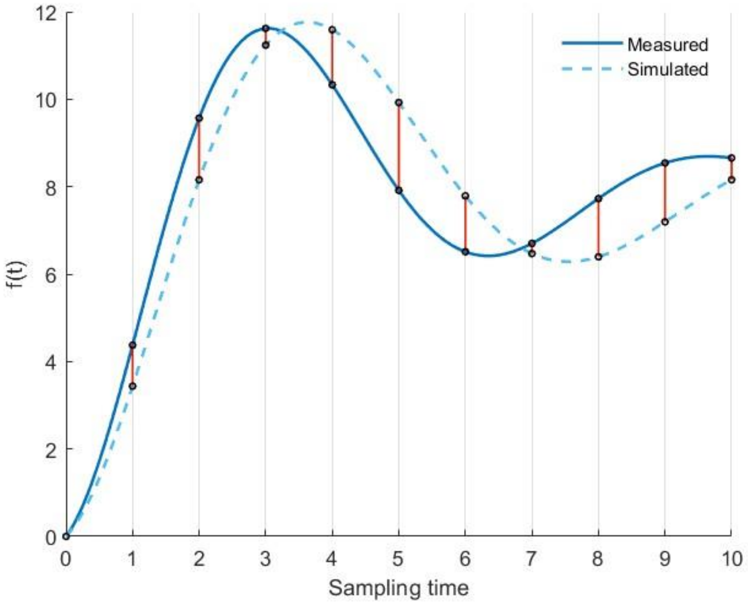


Figure 7 Evaluation of $f_{RSS}(x)$

With the objective of improving the estimation capabilities, different definitions for the objective function were tested, which included the Huber loss and the sum of absolute residuals. From the test performed, none of the mentioned definitions for the objective function produced results that were superior to the standard sum of squared residuals.

Among all the definitions tested, only one approach proved to be effective in improving the convergence of the optimizer. This approach relied on combining the sum of squared residual with a term that reflects the difference in timing and amplitude between the local extrema or a discontinuity of the measured curve and the simulated one.

This type of contribution in the case of a set of N_{peaks} local maxima was defined in *Equation 2.1*.

$$f_{peak}(x) = \frac{1}{N_{peaks}} \sum_{i=1}^{N_{peaks}} a\left(t^*_{peak_i} - t_{peak_i}\right)^2 + b\left(\ f^*_{x^*}\left(t^*_{peak_i}\right) - f_x\left(t_{peak_i}\right)\right)^2 \quad (2.2)$$

Where a and b are scalar values used to balance the contribution of the cost function. With the proper standardization those parameters could be set to 1. $t^*_{peak_i}$ and t_{peak_i} represent the timing of the i-th peak of measured and simulated characteristic respectively. Lastly, $f^*_{x^*}$ and f_x are the functions that describe the observed and simulated curve respectively.

The number of peaks considered varies on a case-to-case basis since not always incrementing the number of points proved to be beneficial since it undermines the initial objective of prioritizing the convergence of a chosen point. This type of cost function is depicted in the following image.

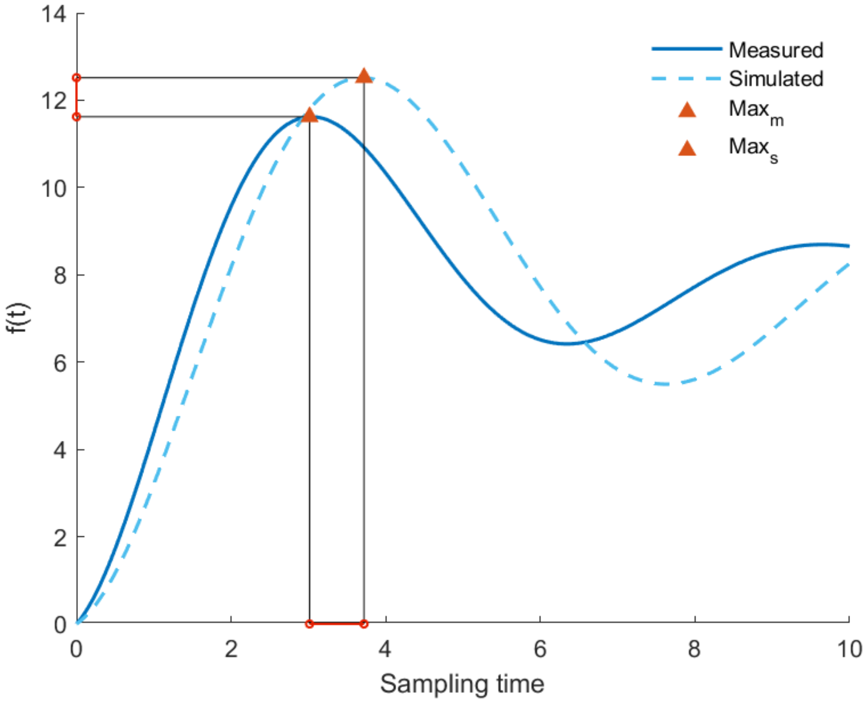


Figure 8 Evaluation of $f_{peak}(x)$

The same definition for the cost function could be applied to all types of critical points, like a local minimum $f_{min}(x)$. The combination of the cost function built on the peak on the valleys of the transitory is defined as $f_{cp}(x)$. The formal definition of $f_{cp}(x)$ is presented in *Equation 2.3*.

$$f_{cp}(x) \ = \frac{1}{2}(f_{peak}\ \ x\ \ + f_{min}\ \ x\ \ ) \tag{2.3}$$

The advantage of incorporating in the cost function a contribution that keeps track of the difference between specific points in the characteristics is that it allows to prioritize the convergence of the aforementioned points, especially in the initial iteration of the estimation when the difference between the two characteristics is the highest. In the tests performed, this resulted in an effective broadening of the range of initial conditions for which the optimizer is able to converge.

From initial tests of the algorithm with the configuration discussed in the paragraph 3.4.1, it was possible to notice that when minimizing $f_{RSS}(x)$, the algorithm was able to converge to the solution x^*, but only if the initial point x was in its proximity. When the difference between x and x^* was too big, the convergence was not guaranteed. The algorithms had the tendency to stagnate in iterations where little to no improvement is achieved. On the other hand, when an optimization algorithm tries to minimize $f_{cp}(x)$, it does not stagnate in parameters that are entirely inexact when more significant perturbations are applied to the initial condition. However, the confidence interval for the identification is larger than in the previous case. This last undesired effect has to be expected since there could be multiple curves that could pass through a limited number of critical points. Nevertheless, the algorithm never failed to converge to the proximity of the solution x^*.

By mapping the two cost functions in the case of the identification of an RLC parallel filter where R and L are two variables, and C is kept constant, it is possible to visualize and compare the two definitions. Although this is a simplified example, it is possible to notice the same enhanced convergence of $f_{cp}(x)$ compared to the $f_{RSS}(x)$ that was observed in tests where the number of parameters to identify was superior.

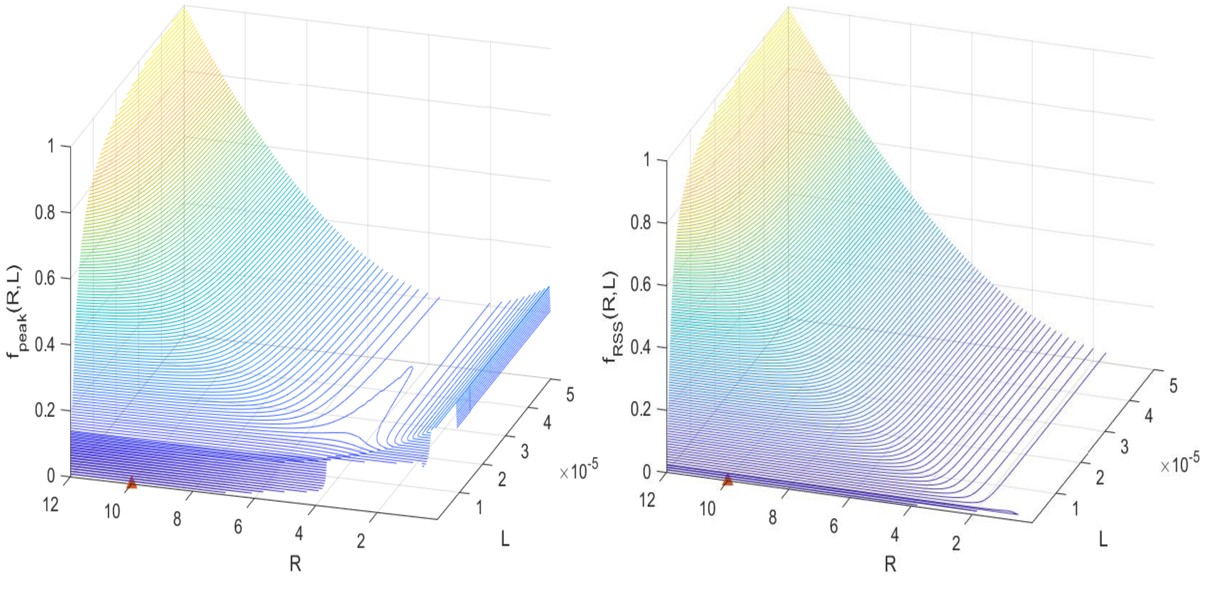


Figure 9 Comparison between f_{cp} (left) and f_{RSS} (right)

Plotting the contour line in the areas where the two functions have the smallest gradient (with the same depth interval in order to have an equal comparison), it is possible to compare the most critical area for the algorithm and highlight the plateau present on $f_{RSS}(x)$. Analyzing the contour line in *Figure 12* it is possible to visualize why $f_{RSS}(x)$ has a lower convergence capability than $f_{cp}(x)$. An Optimizer that implements only $f_{RSS}(x)$ and arrive to a point where the gradient is usually almost zero is unable to exit from that point and find the absolute minimum of the function.

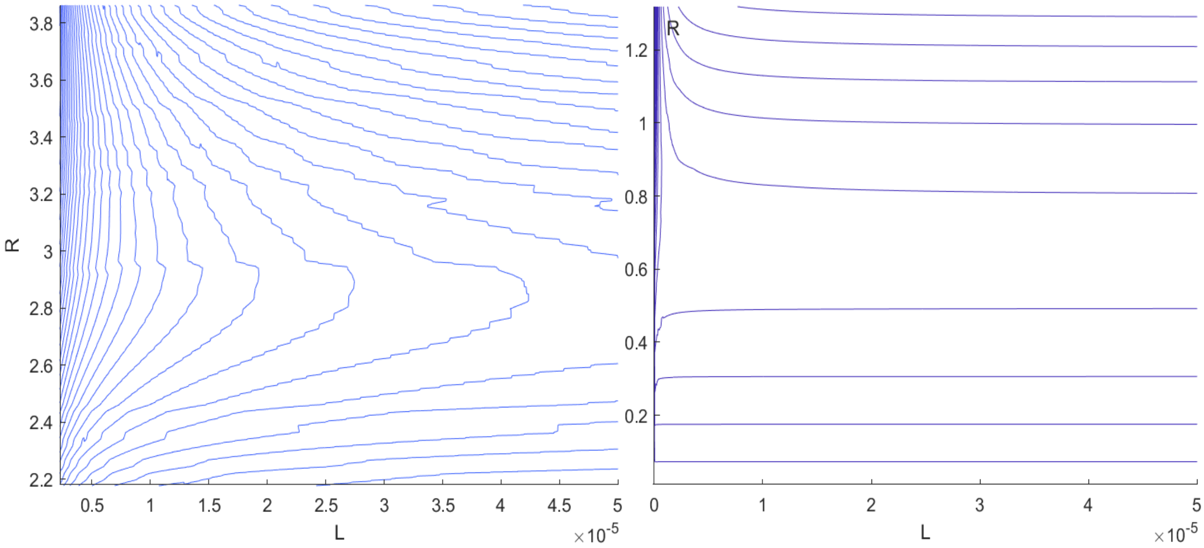


Figure 10 A zoomed comparison between f_{cp} (left) and $f_{RSS}($ (right)

It has to be noted that the minimum magnitude of $\nabla f_{RSS}(x)$ in this mock example is one order of magnitude smaller than the minimum magnitude of $\nabla f_{cp}(x)$. This has the potential to become highly impactful when errors are introduced in the computation of the cost function, which is an inevitable occurrence for more complex cases. The inaccuracies in the evaluation of the cost function could come from:

- **Measurement limitation:** There are inaccuracies in accurately tracking the "target" characteristics due to the unavoidable presence of noise in the measurement and limitations in the sensing hardware.

- **Computation inaccuracies:** Using a model to reproduce the characteristic curve of an electronic device will inevitably introduce an approximation error that comes not only from the finite resolution of the simulation but also from the discrepancies between the model and the real device.

Implementing a cost function that is less sensitive to uncertainties could help achieving a higher degree of robustness in the identification of the parameters, especially in more complex systems where the computational inaccuracies become more significant due to the increased computational burden of the simulation and due to the increased bias in the model.

2.2 The optimizer

In [1], the algorithm used for the optimization was the trust-region reflective non-linear least square present in the standard MATLAB optimization toolbox. Even if this approach solved the estimation problem with a good degree of accuracy (the average R^2 is 0.9863 for the estimated parameter) is a very time-consuming process, e.g., the estimation of the passive component in the simplest circuit, the buck converter, required 44 minutes. Other papers propose different algorithms: [9] and [13] implement the particle swarm algorithm, and [8] implement the Bayesian optimization. In this work, the pattern search algorithm was adopted with the objective of testing its capabilities in improving the efficiency of the estimation by reducing the number of function evaluations required for the estimation of the parameters, with the trivial benefit of lowering the computational burden and estimation time required to finish the identification.

The pattern search methods are a subclass of direct algorithm, in which the minimum of function f is searched without the use of derivates [27][29], which is useful when there are several local optima. However, it is essential when Δf is unavailable, either because it does not exist, or it cannot be accurately estimated due to noise in f or other reasons [28]. Among the first direct search algorithms were the well-known method of Hooke and Jeeves and the simplex algorithm of Nelder and Mead [27].

Given an initial point $x_0 \in \Omega$, the pattern search algorithm at each iteration will generate a set of trial points in Ω whose objective function value is possibly smaller than $f(x_k)$where x_k is the current best feasible solution, called the *incumbent [29].* The trial points lie on the mesh M_k defined by

$$M_k = \{x + \Delta_k^m D_z : x \in V_k, z \in \mathbb{N}^{n\,D}\} \in \mathbb{R}^n \qquad (2.4)$$

Where $V_k \subset \mathbb{R}^n$ is the set of all evaluated points by the start of iteration *k*, $\Delta_k^m \in \mathbb{R}_+$ is the *mesh size parameter* at iteration *k*, and *D* is a matrix in $\mathbb{R}^{n \times n_D}$ composed of n_D directions in $\mathbb{R}^n$ *[29].* The computation of the trial points is made in what is defined as the *poll step.* In the poll step a positive spanning set of direction $D_k \in D$ is chosen to construct the poll set. The poll set P_k is constructed as the neighboring mesh points in each of the directions in $\boldsymbol{D_k}$ [27], *w*here the distance from the incumbent is bounded with a scalar parameter $\boldsymbol{\Delta_k^p}$ called *poll size paramete*r, which could be equal to $\boldsymbol{\Delta_k^m}$ or linked through the *mesh index* $\boldsymbol{l_k}$ depending on the specific algorithm[29].

$$P_k = \quad \{x_k + \Delta_k^m d : d \in D_k\} \subset M_k \tag{2.5}$$

The function *f* is then computed in $\boldsymbol{P_k}$ until all the points are evaluated, or a point with a lower objective function value is found [27]. If the poll step successfully finds an improved mesh point, it becomes the new incumbent $\boldsymbol{x_{k+1}}$, and the mesh is coarsened [27]. Otherwise, if the poll step fails to find an improved mesh point, the incumbent is a mesh local optimizer and remains unchanged [27]. To further improve the incumbent, the mesh has to be refined.

More advanced algorithms add a preliminary step to the poll, the search step. In the search step, the objective function *f* is evaluated at a finite number of points lying on the current mesh in an attempt to find a new point with a better function value [29]. In the search step, user knowledge about the problem coupled with inexpensive surrogates for the objective function or constraints could be used. The surrogate functions are typically evaluated at several mesh points, and the expensive functions are then evaluated only at the most promising trial points [27].

The *patternsearch* function in MATLAB offers a plethora of different polling and search algorithms. The performance of all those options may vary depending on the problem the user is trying to solve [27].

The type of poll methods that are present in the MATLAB library are:

- GPS (Generalized Pattern Search)
- GSS (Generalized Set Search)
- MADS (Mesh Adaptive Direct Search)

Each of the poll methods listed is used in several algorithms that, while implementing the same general approach, differ in the specific implementation of the method.
Secondly, patternsearch has the following seven search methods:

- GPS (Generalized Pattern Search)
- GSS (Generalized Set Search)
- MADS (Mesh Adaptive Direct Search)
- searchga (search using Genetic Algorithm)
- searchlhs (search using Latin Hypercube Algorithm)
- searchneldermead (search with Nelder-Mead algorithm)
- rbfsurrogate (search with radial basis function surrogate)

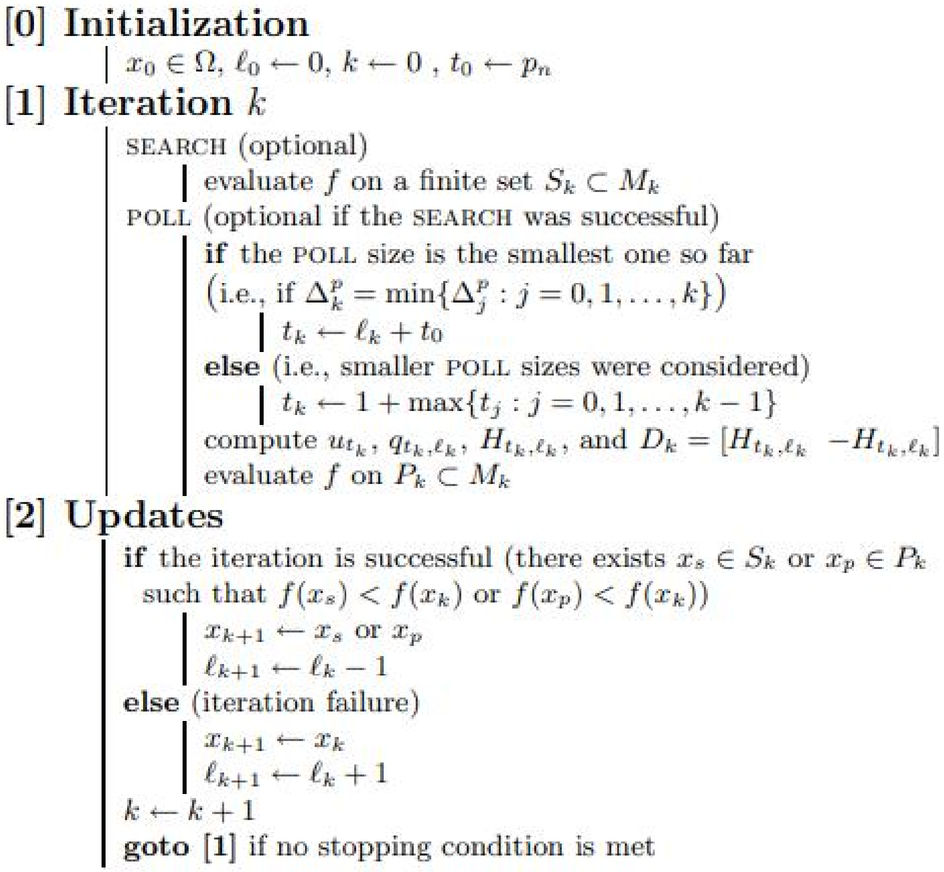
A detailed description of these algorithms is given in MathWorks Global Optimization Toolbox User's Guide. It has to be noted that if the poll and search steps share the same methods, the poll step is skipped since the results would be identical [27].

The class of algorithm that had the most success in solving the problem under consideration was the Mesh Adaptive Direct Search. Implementing a MADS algorithm for both the search step and poll step significantly impacted the optimizer's efficiency in minimizing the objective function. The two algorithms implemented for the search and poll step were *"OrthoMADSPositiveBasisNp1"* and *"MADSPositiveBasis2N",* respectively. Specifically, the *"OrthoMADS"* algorithm proved to be particularly suited for the task and consistently solved the problems efficiently.

OrthoMads algorithm implements a deterministic method to generate the poll direction. This method generates an orthogonal basis by implementing the quasi-random Halton sequence [29]. As described in detail in [29], the main steps implemented by the algorithm to construct the directions D_k are the following:

- A quasi-random Halton sequence is implemented to produce a vector in $[0,1]_n$.
- The vector generated is then scaled and rounded to an appropriate length. The resulting direction is called the *adjusted Halton direction.*
- The Householder transformation is applied to the adjusted Halton direction, producing n orthogonal and integer vectors, forming a basis for $\mathbb{R}^n$
- The basis is completed to a positive basis formed by $2n$ OrthoMads poll directions D_k, by including in D_k the basis and its negatives.

Finally, *Figure 13* represents all the operational steps previously described that compose the OrthoMADS algorithm.



```
[0] Initialization
    $x_0 \in \Omega$, $\ell_0 \leftarrow 0$, $k \leftarrow 0$ , $t_0 \leftarrow p_n$
[1] Iteration $k$
    SEARCH (optional)
        evaluate $f$ on a finite set $S_k \subset M_k$
    POLL (optional if the SEARCH was successful)
        if the POLL size is the smallest one so far
        (i.e., if $\Delta_k^p = \min\{\Delta_j^p : j = 0, 1, \dots, k\}$)
            $t_k \leftarrow \ell_k + t_0$
        else (i.e., smaller POLL sizes were considered)
            $t_k \leftarrow 1 + \max\{t_j : j = 0, 1, \dots, k-1\}$
        compute $u_{t_k}$, $q_{t_k,\ell_k}$, $H_{t_k,\ell_k}$, and $D_k = [H_{t_k,\ell_k} \;\; -H_{t_k,\ell_k}]$
        evaluate $f$ on $P_k \subset M_k$
[2] Updates
    if the iteration is successful (there exists $x_s \in S_k$ or $x_p \in P_k$
     such that $f(x_s) < f(x_k)$ or $f(x_p) < f(x_k)$)
        $x_{k+1} \leftarrow x_s$ or $x_p$
        $\ell_{k+1} \leftarrow \ell_k - 1$
    else (iteration failure)
        $x_{k+1} \leftarrow x_k$
        $\ell_{k+1} \leftarrow \ell_k + 1$
    $k \leftarrow k + 1$
    goto [1] if no stopping condition is met
```

Figure 11 The OrthoMADS algorithm, figure from [29]

2.3 Performance indices

In order to benchmark the performance of different algorithms, it is required to provide an estimate not only of the efficiency of the optimization algorithm but also of the quality of its output. The efficiency of an optimization algorithm is usually referred to as the computational effort required to obtain a solution. The main parameters used to define the computation effort are the number of function evaluations and running time. Another value that could be used is memory but is less common [30].

In complex circuits, the step that had the highest computational burden was the simulation of the model in order to evaluate the cost function. The time required for the algorithm to compute the next evaluation point is, in most cases, negligible when compared to the time required to compute the cost function. The only exception encountered was with the *bayesopt* algorithm. For this reason, the parameter used to evaluate the algorithm's efficiency was the number of function evaluations.

Providing an indicator for the quality of the algorithm output is less intuitive, and there are a variety of methods. The condition that the approaches to measure the quality of the solution is the availability of the solution [30]. For the purpose of the analysis, the solution was considered available, and in particular, a fixed cost method was implemented where the final optimization error is the value used to define the quality of the output. In order to clearly display the result of the simulation, two profile methods were implemented: the accuracy profile and the data profile.

Both of these profile methods allow to abridge the information about the quality and efficiency of a solver $s \in S$ in solving batch of problems $p \in P$. Each optimization problem p is characterized by a unique starting point x_p^0, but the same solution x_p^*.

2.3.1 Accuracy profile

In the accuracy profile method, the ratio between the problem solved by the algorithm and the relative accuracy is depicted [31]. This profile method provides a tool to analyze the output quality of the entire optimization benchmarking test. First, an accuracy measure is defined for each problem $p \in P$ and each solver $s \in S$.

$$\gamma_{p,s} = \begin{cases} -f_{acc}^{p,s} & if\ -f_{acc}^{p,s} \leq M \\ M & if\ -f_{acc}^{p,s} > M \quad or\ -f_{acc}^{p,s}\ is\ undefined \end{cases} \tag{2.6}$$

Where $f_{acc}^{p,s}$ was defined as:

$$f_{acc}^{p,s} = \log_{10}\left(\sqrt{\frac{\left\|f(\bar{x}_{p,s}) - f(x_p^*)\right\|_2^2}{n_{points}}}\right) \tag{2.7}$$

In this equation, $x_{p,s}$ represent the solution proposed by the solver s to the problem p and $f\ x$ is the transitory response of the circuit with parameters equal to the vector x. Instead of evaluating the quality of the identification on the cost function values, the RMSE between the transitory curve in time with the real parameter x_p^* and the identified parameter $x_{p,s}$ was chosen as the indicator for the accuracy of the output. This decision was made in order to have a consistent index through all the testing since different types of cost functions were tested. In the following plot, an example of an accuracy profile is depicted in *Figure 14*.

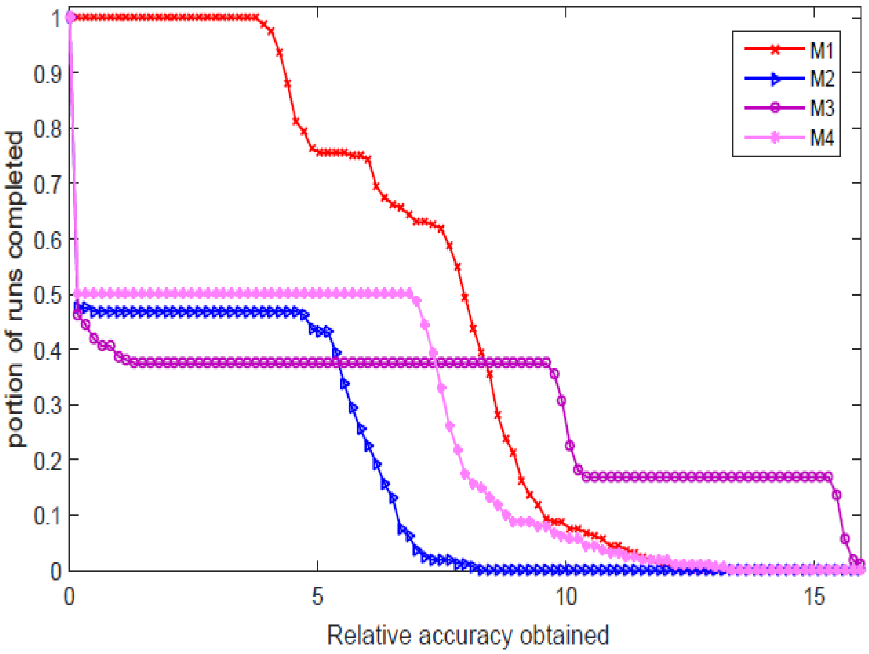


Figure 12 Example of accuracy profile, figure from [30]

By examining the accuracy profile of this test case, it is possible to make the following consideration: the M1 method achieves a five-digit accuracy in almost all the problems under examination. Meanwhile, all the other methods only achieve this level of accuracy for a maximum of 50% of the test problems. Therefore, M1 is the method to adopt when the project requires solving the highest number of problems, and a five-digit accuracy is deemed sufficient. Method M3 achieved a twelve-digit accuracy on almost 40% of the test problems, and almost half of those problems achieved a fifteen-digit accuracy. This level of accuracy is not reached by a significant portion of the test problem in any of the other methods. Therefore, M3 is the method that has to be implemented when the requirement for accuracy is more stringent.

2.3.2 Data profile

More and Wild introduced data profiles in [32] to benchmark derivative-free optimization algorithms. This profile method is used to display the efficiency of an optimizer in solving a problem. The profile displays the percentage of problems (for a given tolerance τ) that can be solved within a computational budget α [30]. To represent the computational budget, the number of function evaluations was used. The data profile of an optimization algorithm s is defined using the following equation:

$$d_s\ \alpha\ = \frac{1}{|P|} size\{p \in P: t_{p,s}\ \le \alpha\} \tag{2.8}$$

In which $t_{p,s}$ represents the number of function evaluations required to satisfy the convergence test, and $d_s \;\; \alpha$ is the percentage of problems that can be solved within α function evaluation. An example of a data profile graph is presented in *Figure 15.*

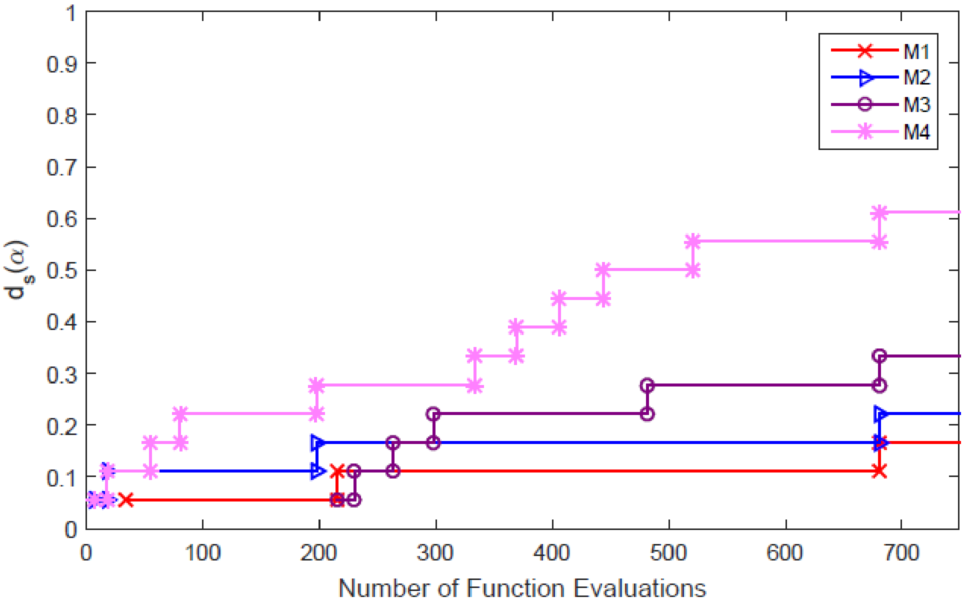


Figure 13 Example of data profile, figure from [30]

From *Figure 15*, it emerges that method M4 is the most efficient in solving the test problems. Other considerations could also be made for the other solvers depending on the function evaluation budget available in the requirement. It has to be noted that Data profiles do not provide the number of function evaluations required to solve a specific problem but instead provide a visualization of the aggregate data [30].

Chapter 3

Test on an RLC filter

3.1 The models

Before implementing the parameter estimation on a complex converter, a simple RLC filter was used as a proof of concept. Implementing a mock-up case allowed to have a simpler environment to test and develop different solutions for the code-related challenges that will inevitably arise.

The first step was to choose a topology for the RLC that has to be identified and then compute the dynamical equation that governs its behaviors. The model implemented, and the computation of its dynamical equations are presented below.

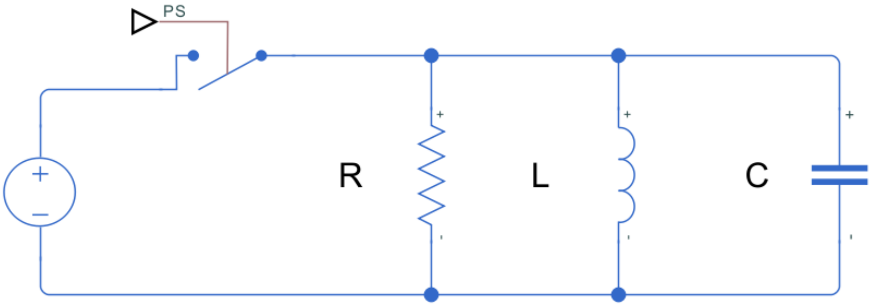


Figure 14 Schematics of the first model

This model is composed of two active components and therefore two state equations are required to fully describe the behavior of this model. By applying the first Kirchhoff law on the node:

$$-I_0 + \frac{v_c}{R} + i_L + C\frac{d}{dt}v_C = 0 \tag{3.1}$$

Then, considering the equation of the impedance

$$v_L = v_C = L\frac{d}{dt}i_L \tag{3.2}$$

Translating the equations in the matrix form to implement them in MATLAB.

$$\begin{bmatrix} -C & 0 \\ 0 & L \end{bmatrix} \frac{d}{dt} \begin{bmatrix} v_c \\ i_L \end{bmatrix} = \begin{bmatrix} 1/R & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v_c \\ i_L \end{bmatrix} - \begin{bmatrix} I_0 \\ 0 \end{bmatrix} \tag{3.3}$$

For the first tests, the model used to identify the parameters was equal to the *"target"* model that had to be identified. In order to verify the robustness of the identification, a variety of tests were performed with the introduction of noise in the *"target"* characteristics and later with the introduction of some parasitic component to create a discrepancy between the model used in the optimizer for the identification and the *"target"*, thus allowing to study how the parameter estimation algorithm behaves in the case of an asymmetry between the real circuit and the model used for the identification. A series resistor was added to the inductive and capacitive components to introduce a bias between the two models, thus allowing to induce an unbridgeable difference between the two characteristics without changing the topology of the circuit. The second model and the computation of its differential equation are presented below.

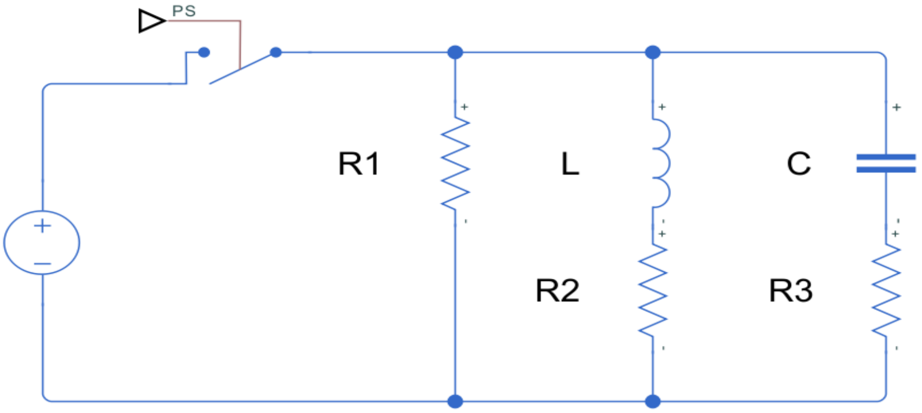


Figure 15 Schematics of the second model

Even for this model, there are only two state variables, and therefore, only two differential equations are required to describe this model. Implementing the first Kirchhoff law on the node:

$$I_0 = \frac{V_{R1}}{R_1} + I_L + C\frac{dV_c}{dt} \tag{3.4}$$

By expanding V_{R_1} as the combination of the voltage drop across the resistor R_3 and the capacitor is possible to find the first equation:

$$I_0 = \frac{1}{R_1}\left(V_C + R_3 C\frac{dV_c}{dt}\right) + I_L + C\frac{dV_c}{dt} \tag{3.5}$$

Similarly, the second equation is derived from the balance of voltages:

$$V_{R_1} = L\frac{dI_L}{dt} + R_2 I_L = R_1 I_1 = R_1\left(I_0 - I_L - C\frac{dV_c}{dt}\right) \tag{3.6}$$

With some simplification it becomes.

$$\frac{L}{R_1}\frac{dI_L}{dt} + C\frac{dV_c}{dt} = I_o - I_L - \frac{R_2}{R_1}I_L \tag{3.7}$$

By rearranging all the terms, the state equations in matrix form are:

$$\begin{bmatrix} -C\left(\dfrac{R_3}{R_1}+1\right) & 0 \\ -C & -\dfrac{L}{R_1} \end{bmatrix} \frac{d}{dt}\begin{bmatrix} v_c \\ i_L \end{bmatrix} = \begin{bmatrix} \dfrac{1}{R_1} & 1 \\ 1 & \left(1+\dfrac{R_2}{R_1}\right) \end{bmatrix} \begin{bmatrix} v_c \\ i_L \end{bmatrix} - \begin{bmatrix} I_0 \\ I_0 \end{bmatrix} \tag{3.8}$$

Once the equations that describe both models were found, the transient response caused by the input was computed. The transient response that will be implemented in the identification process is induced by a simple step function as the input. For consistency, the same input will be used for all the models and tests that were performed.

3.2 The cost function

The first step in building the cost function was to choose which variable to measure in order to have the best identification. In this case, the choice was between the voltage of the capacitor or the current of the inductor. For this analysis, the variable used in the identification was the current, since it was the variable that saw the highest variation in its value when changing the system parameter. The step response that characterizes the model is composed by a series of oscillations. The first local maxima and minima are especially prevalent in the "target" characteristic. For this reason, those two points were chosen to construct the $f_{cp}(x)$ that is then added to the FRSS, as discussed in the previous chapter. In order to balance the timing and amplitude contributions of the $f_{cp}(x)$, both the time and amplitude of the characteristic were normalized with respect to their respective maximum values.

Optimizing the ratio between those different elements of the cost function is, therefore, mandatory to optimize the efficiency of the identification. In order to choose a ratio between all the different components of the cost function, a series of benchmarks were performed. Implementing the accuracy and data profiles, it was possible to compare which ratio improved the efficiency of the optimizer in solving the problem. The ratios were tuned by gradually increasing the contribution of $f_{cp}(x)$ until the identification quality started to be negatively impacted by this process. The accuracy of the identification did not suffer initially, but for a more significant contribution of $f_{cp}(x)$, a degradation of the final resolution was observed. Meanwhile, by increasing the contribution of $f_{cp}(x)$, the portion of problems that were solved from the algorithm initially saw an increase, but after a while, it reached a point of diminishing returns, after which any increase of $f_{cp}(x)$ started to hinder the ability of the algorithm to converge. The end product of this tuning process, is a cost function defined as:

$$f_{Cost}\ x\ = 0.4f_{RSS}\ x\ + 0.3f_{peak}\ x\ + 0.3f_{valley}(x) \tag{3.9}$$

The result of the tuning is not the best possible mix of the different contributions for the specific identification problem but one that significantly improves the result obtained with only the $f_{RSS}\ x$. This was a design choice since the performance of a specialized cost function for a solution $\boldsymbol{x}^*$ will never be matched once the solution of the identification starts to drift due to temperature shifts and the ageing of the device.

3.3 Defining the tests problems

The benchmark of the different algorithms was made by generating a set of problems P. Each problem p is characterized by the same solution x^* and defines a unique perturbation of the initial conditions x_{0p}. Once the test set is defined, it must be used in all of the different tests in order to yield meaningful results from the comparison that has been made [30]. Poorly designed tests in the benchmark will incur the risk of cloaking the weaknesses or strengths of an optimization algorithm. It is therefore crucial, in order to avoid misleading results, that an appropriate test set should generally seek to avoid the following deficiencies [30].

- **Not enough problems**. Trivially, a limited number of tests will inevitability impact the reliability of the findings.

- **Biased starting points**. Not implementing the same set of problems will unavoidably bias the results. However, more subtle problems can also exist. For example, if a starting point lies on the boundary of a constraint set, then an interior point method will be severely disadvantaged [30].

- **Too little variety in the problem difficulty**. Simply increasing the number of problems tested will not always result in meaningful findings. Especially when the computational burden of the test becomes non-negligible, a particular emphasis on defining the problem set in order to efficiently test the algorithm for a variety of difficulties is required.

The problem set was built by following those guidelines. In [30], the minimum number of tests required to compare different versions of an optimization algorithm effectively is in the order of one hundred. Following this guideline, when defining P, it was decided to more than double the minimal recommendation for the number of problems and have $dim(P) = 250$.

Implementing a relatively small problem set requires thoughtfully choosing how the problem set is generated in order to diversify the problems' difficulties and produce the most meaningful data with the least amount of testing. For those reasons, a parameter that could be linked to the difficulty of a problem was required when generating a test set.

The parameter that was chosen to represent the difficulty of a problem was the root mean square error between the characteristic response of the solution x^* and the one of the problems x_p that was defined as RMSEic. It is an index used to represent the difference between the characteristics that the algorithm will have to solve and the solution.

The choice of RMSEic as the index for the difficulty of a problem was made because, from the different tests performed, it was the parameter that was the most strongly related to the number of iterations required to solve the problem, and its value was also a reliable index to predict and assess the convergence of the algorithm.

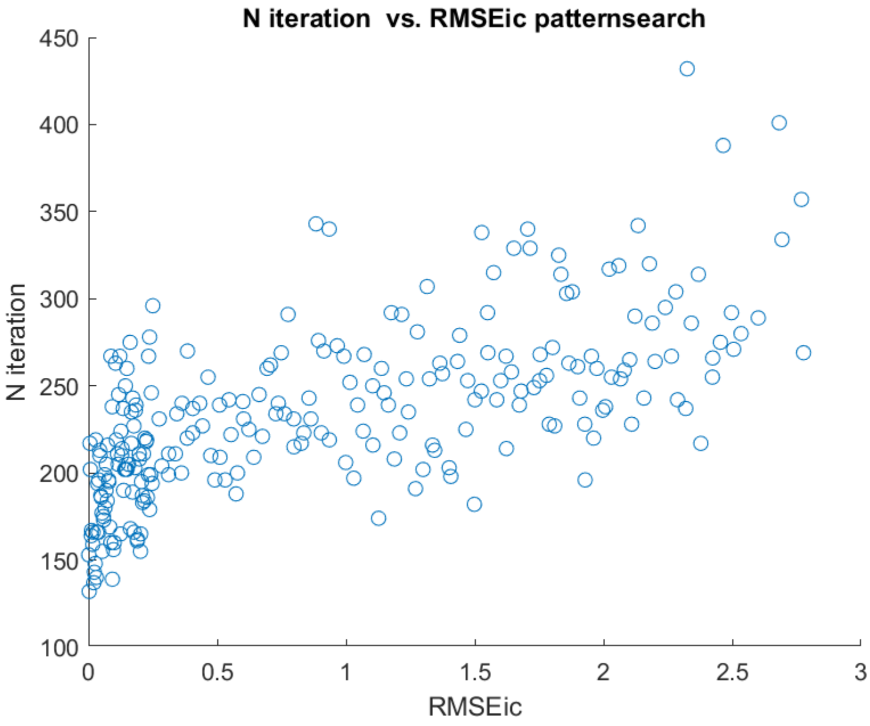


Figure 16 N of iteration as a function of the initial RMSE

To introduce the widest variety in the problem difficulties, the objective in the generation of the problem set was to have a uniformly distributed RMSEic that spans from 0 to an upper boundary that must be sufficiently high in order to encompass the initial condition from which the algorithm is unable to converge.

The upper boundary for the RMSEic was chosen by studying the convergence capabilities of the patternsearch algorithm in the ideal case where neither bias nor noise is present in the model. Following the guidelines in [30], it was purposely set high enough to include in the set of problems P a portion of around 20% of problems that the patternsearch algorithm was unable to solve. It was possible to implement the RMSEic since the convergence of the patternsearch algorithm is characterized by a threshold RMSEic, after which the conversion probabilities are drastically reduced, as shown in the following picture.

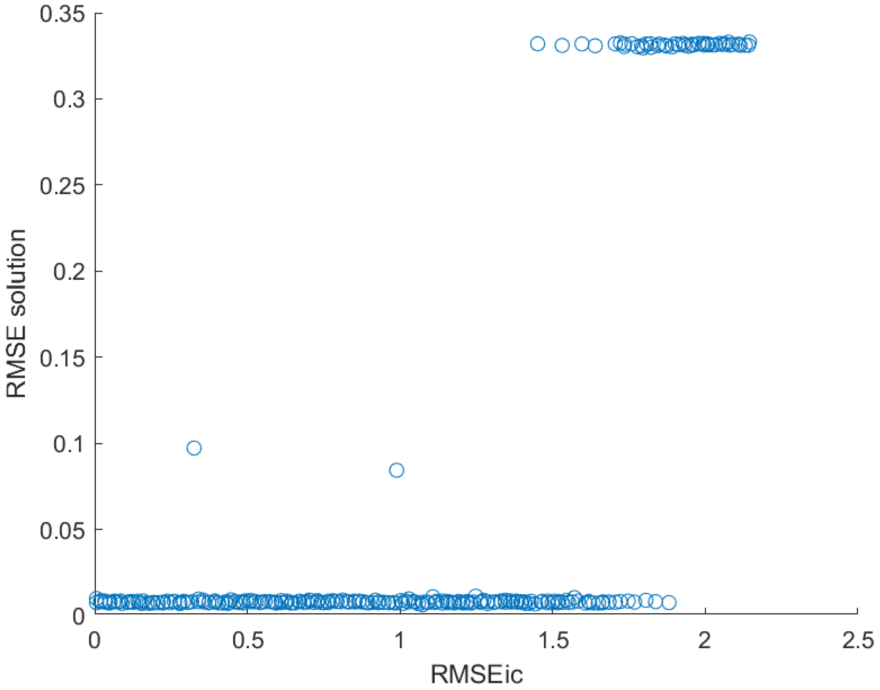


Figure 17 Scatter plot between the RMSEic and the RMSE of the solution found by the optimizer.

This graph is characterized by two main levels for the RMSE of the solutions, the presence of a discrete level in the RMSE solution might suggest the presence of a local minimum where all the problems that with higher RMSE will converge. This hypothesis was tested and proved to be untrue.

The problems that the algorithm is unable to solve do not converge to the same combination of parameters. The similarity in the RMSE is the effect of the topology of the RMSE as a function of the parameters, which present the same plateaus that were visualized in the simplified case of f_{RSS}. The behavior of the output RMSE allowed to define a threshold to discern between the problems that the algorithm was able to solve within an acceptable margin of error from the ones that did not meet the desired standard of accuracy.

Since the simulated time used to generate the characteristic is fixed, a boundary on the position of the first peak must be implemented. Otherwise, there is the risk that the randomly generated initial condition will be characterized by an extremely large time constant, thus resulting in a transitory response that, in the time frame of the simulation, could be approximated to a constant. This situation is an extremely unfavorable one since the algorithm is incapacitated to detect the consequences of the variation of a parameter and, therefore, consistently unable to converge. The position of the peak was computed using the following equations.

$$\omega_n = \frac{1}{\sqrt{LC}} \tag{3.10}$$

$$\zeta = \frac{\sqrt{L}}{2R\sqrt{C}} \tag{3.11}$$

$$T_{peak} = \frac{\pi}{2\omega_n\sqrt{1-\zeta^2}} \tag{3.12}$$

To achieve all those requirements, the MATLAB script implemented followed the following algorithm.

1. **Generation of the point** the point is generated by perturbating the solution x^* with a uniform distribution bounded between $[-ex^*\ , ex^*]$ where e is a scalar.

2. **Check feasibility of the point** When a point is considered feasible, it can go to the next step. In the contrary case, it will be discarded. The point is considered feasible if it respects the following conditions:
   - It respects the boundaries.
   - The first peak occurs in the time limit of the simulation $T_{peak} < T_{sim}$

3. **Compute the RMSEic** Use a function to compute the RMSE between the characteristics of the solution x^* and the perturbed point.

4. **Validate the point** If there are no other points with similar RMSEic, the point becomes part of the test set. Otherwise, it will be discarded. After a set number of discarded points, the scalar e that control the magnitude of the perturbation is increased.

This algorithm progressively increases the magnitude of the perturbation, allowing the generation of a set with an increasingly larger RMSEic. This method was chosen since it proved to be an efficient solution in the generation of the test set.

3.4 Identification results

The performance of the identification was tested in four different cases: with and without the parasitic component in the model and with and without noise to disturb the input characteristics. In order to make some comparisons with the patternsearch algorithm, the non-linear least square was chosen to generate the reference performance. The non-linear least square was chosen since it was the algorithm used in [1], which is the paper from which the approach to the identification was taken.

As in [1], the cost function that the non-linear algorithm will try to minimize is a simple sum of squared residuals. This will showcase the eventual advantages of implementing a cost function that is composed of different contributions, allowing us to assess the advantages of this method. In further tests, there was an attempt to adopt the same cost function used in the patternsearch algorithm, but to no avail since the algorithm did not recognize the function as a sum of squares.

3.4.1 Test case without noise or bias in the model

Initially, the algorithm was tested in the simplest of cases where there was no noise to perturb the "target" characteristic, and the model used for the identification perfectly matched the one to identify. Even if this scenario did not present any disturbances, it was necessary to define a baseline performance of the algorithms, allowing to track the eventual degradation of the identification once other factors like noise and bias are introduced. The parameter of the model for this initial baseline are presented in Table *3.*

R [Ω]	L [nH]	C [nF]
10	100	1

Table 3 System parameter

The transitory response analyzed in the different tests was generated by injecting an impulsive voltage of 5V at the capacitor ends. As previously discussed, the cost function used in the identification was built on the transitory of the inductor current.

All the identification data were then condensed to the previously discussed data profile and accuracy profile graph in order to highlight the limitations of the different algorithms in solving the problem. By analyzing the data profile presented in *Figure 20*, it is possible to observe the enhanced capability of conversion of the patternsearch algorithm; not only is it able to converge to a solution with less iteration, but it is also able to converge to a more significant portion of the problem presented.

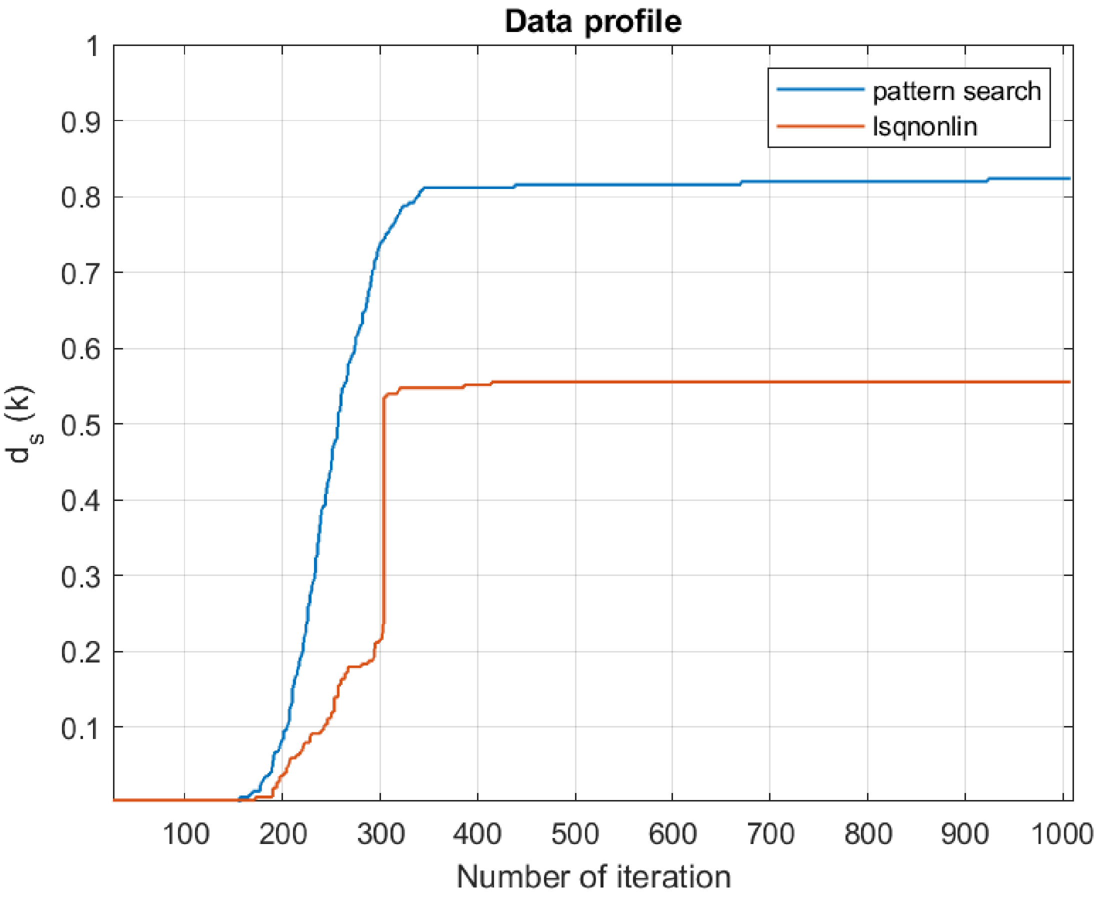


Figure 18 Data profile comparison of patternsearch and lsqnonlin

It has to be noted that the fact that the patternsearch algorithm converged to only 80% of the problem set P. This was a design choice, in the generation of the problems in order to test the convergence capabilities to the max, as recommended in [30] a portion of the problem set P should not be solvable by the algorithm.

By analyzing now, the accuracy profile of the solution presented in *Figure 21*, comparing the patternsearch and the non-linear least square algorithm, it is possible to notice that in this first test case, the solution found by the non-linear least square algorithm had a higher degree of accuracy when compared with the solution of the patternsearch algorithm. From the accuracy profile it emerged that the patternsearch algorithm can solve 100% of the problems with an RMSE that is zero until the third significant figure while the non-linear least square can solve less problems but 100% of the solved problems are solved with an RMSE that is zero until the fifth significant figure.

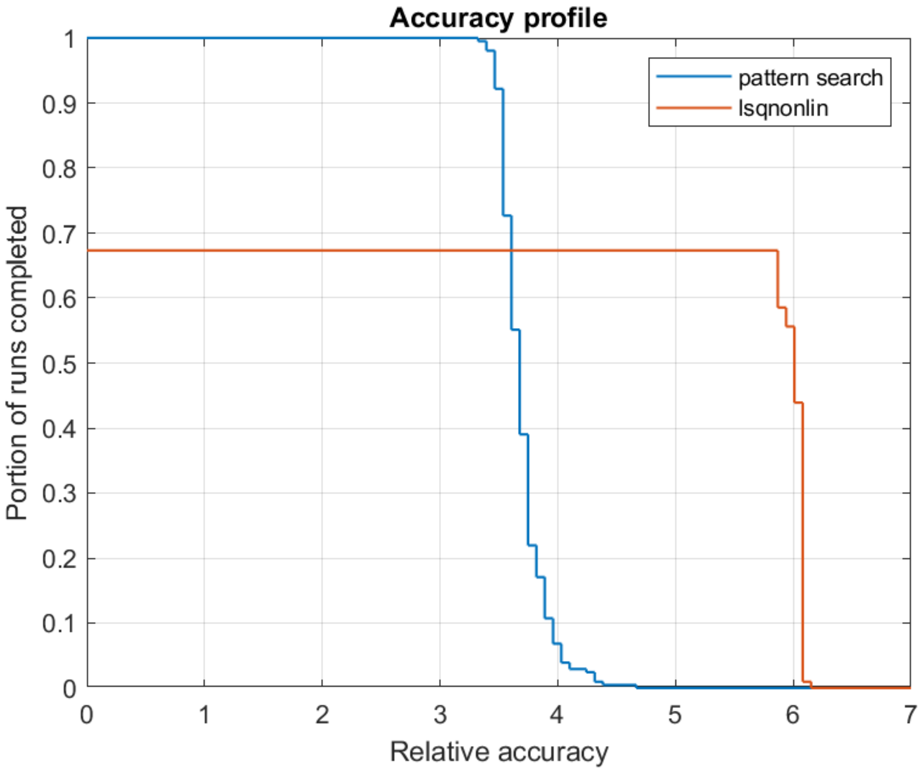


Figure 19 Accuracy profile comparison of patternsearch and lsqnonlin

3.4.2 Test case with noise and without bias in the model

As in the previous test the transitory response was generated by a 5V impulse on the capacitor ends. The system parameters were unchanged from the previous test and are the ones presented on *Table 3*. For this test, the “target” characteristic was perturbed with white noise. The noise-to-signal ratio used for this test was 20db. Before proceeding with the identification process, the "target" characteristic was filtered with a simple low pass filter in order to remove the high frequencies that would disrupt the cost function evaluation. The graph presented in *Figure 22*, represents the transitory of the inductor current measured in Ampere over time expressed in second. In *Figure 22* are present both the perturbed characteristics that was the input of the algorithm and the identified characteristics with the estimated parameter.

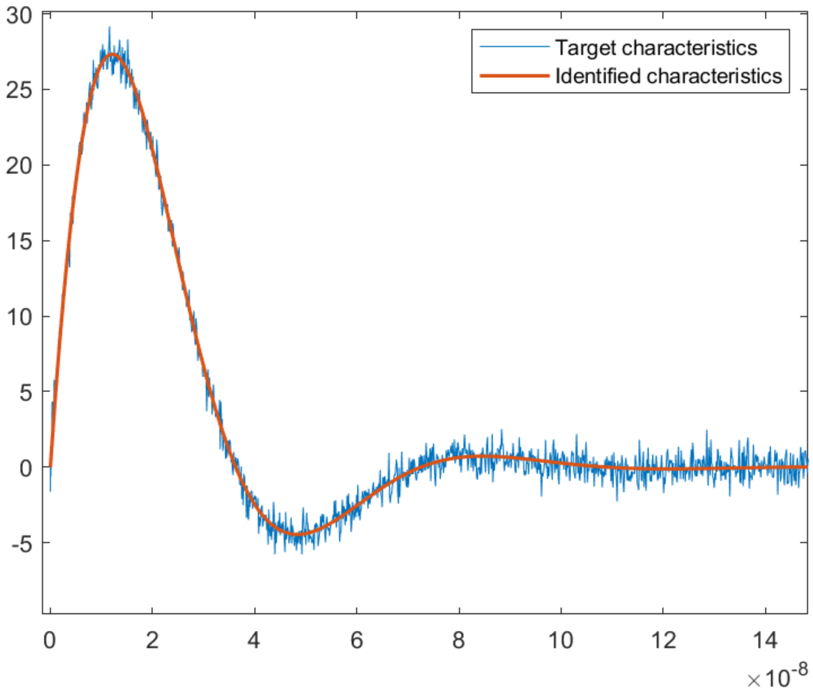


Figure 20 Comparison between the input and the identified characteristic

The parameters that were identified with the patternsearch algorithm were then organized and presented in the distribution histograms of *Figure 23*.

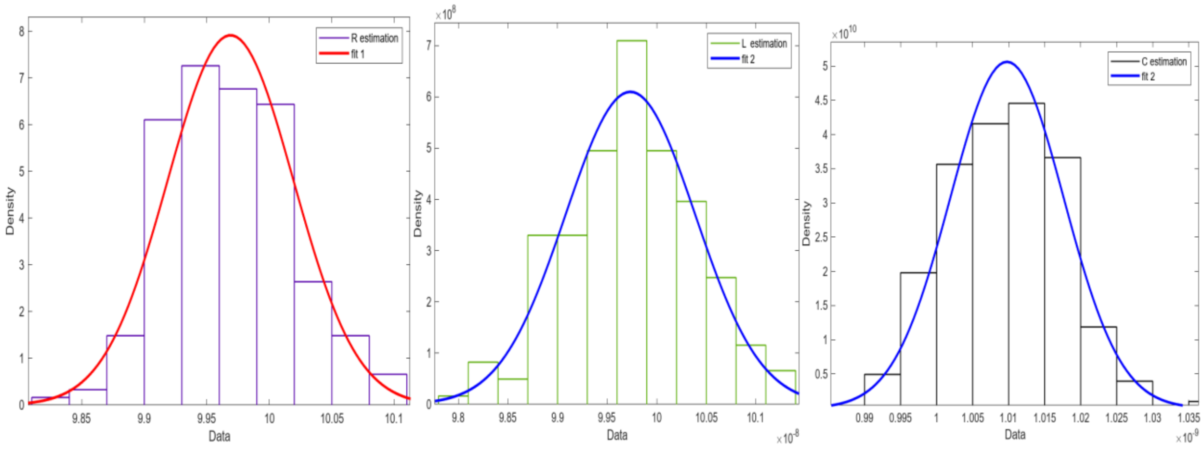


Figure 21 Distribution of the identified parameter: R (left), L (center) and C (right)

In order to assess the quality of the estimation, a normal distribution was fitted to the frequency distribution of the parameters found. The mean and variance of those normal distributions are reported in *Table 3*.

	R [Ω]	*L [nH]*	*C [nF]*
Mean	9.96898	99.7346	1.00977
Variance	0.0025411	4.27882e-19	6.20942e-23

Table 4 Mean and variance of the normal distribution fitted to the estimated results.

Observing the result obtained with the patternsearch algorithm, it is possible to notice that the variance of the fitted Gaussian distribution is minimal, and the most significant deviation of the mean value from the solution is around 0.04% in the resistor.
By analyzing the data profile under this condition, the difference between the characteristics of the two algorithms has enlarged. The convergence characteristics of the patternsearch algorithm did not change drastically when compared with the baseline test without noise or bias to disturb the measured characteristic. Most of the problems were solved within the 300-function evaluation, after which there is a point of diminishing return. On the other hand, the presence of noise significantly worsened the performance of the non-linear least square. The final portion of problems solved by the non-linear least square was 54%, which is not dissimilar from the result in the baseline test, where 56% of the test set P was solved. The number of function evaluations required to solve the problem set was higher when compared with the baseline.

The data profile obtained for this test is presented in *Figure 24*. When compared with the one obtained for the previous test in *Figure 20*, it is possible to notice that even if the overall number of problems solved is comparable for both algorithms, the number of function evaluations required is increased. The algorithm that was more harshly affected by the presence of noise in the signal was the non-linear least square. In the previous test, the non-linear least square was able to solve most of the problems within the 320 iterations, while in this test most of the problems required at least 500 function evaluations.

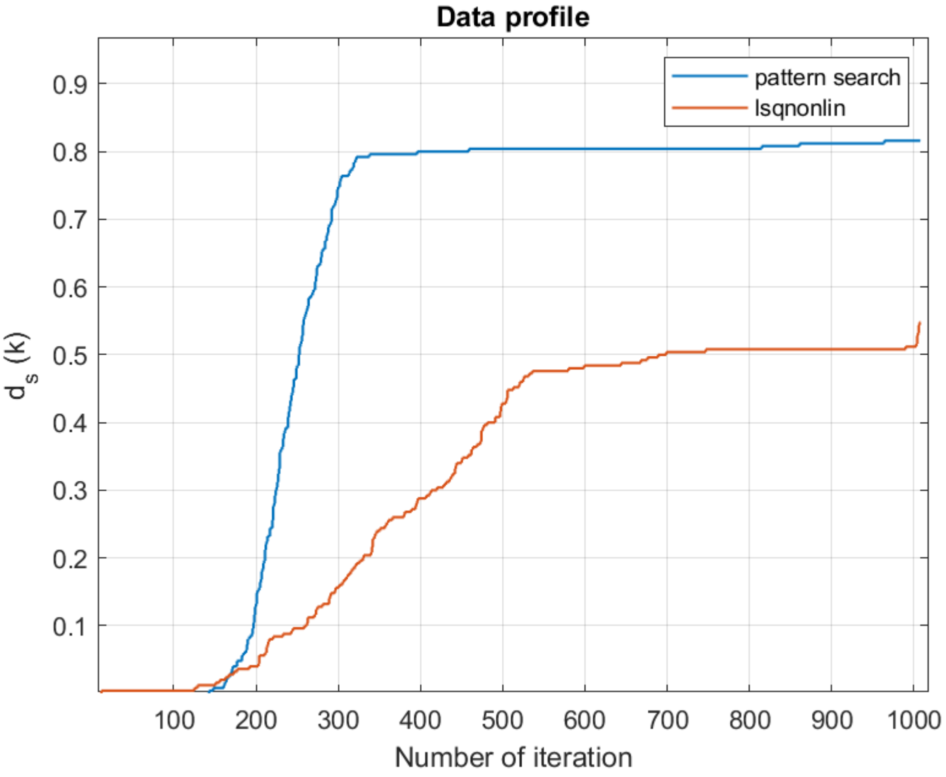


Figure 22 Data profile comparison of patternsearch and lsqnonlin with noise

The accuracy profile obtained in this test is presented in the *Figure 25*. From the accuracy profile of the patternsearch algorithm, it is possible to observe an expected degradation in the quality of the solutions with respect to the baseline test without noise. The accuracy profile, in this case, was characterized by the same shape present on the baseline test but shifted to the left of one unit. The sharp fall in the curve indicates that the RMSE of the patternsearch solutions are consistent throughout the test set. The shift in the curve indicates a rise in the average RMSE of the solutions of a factor of around ten.

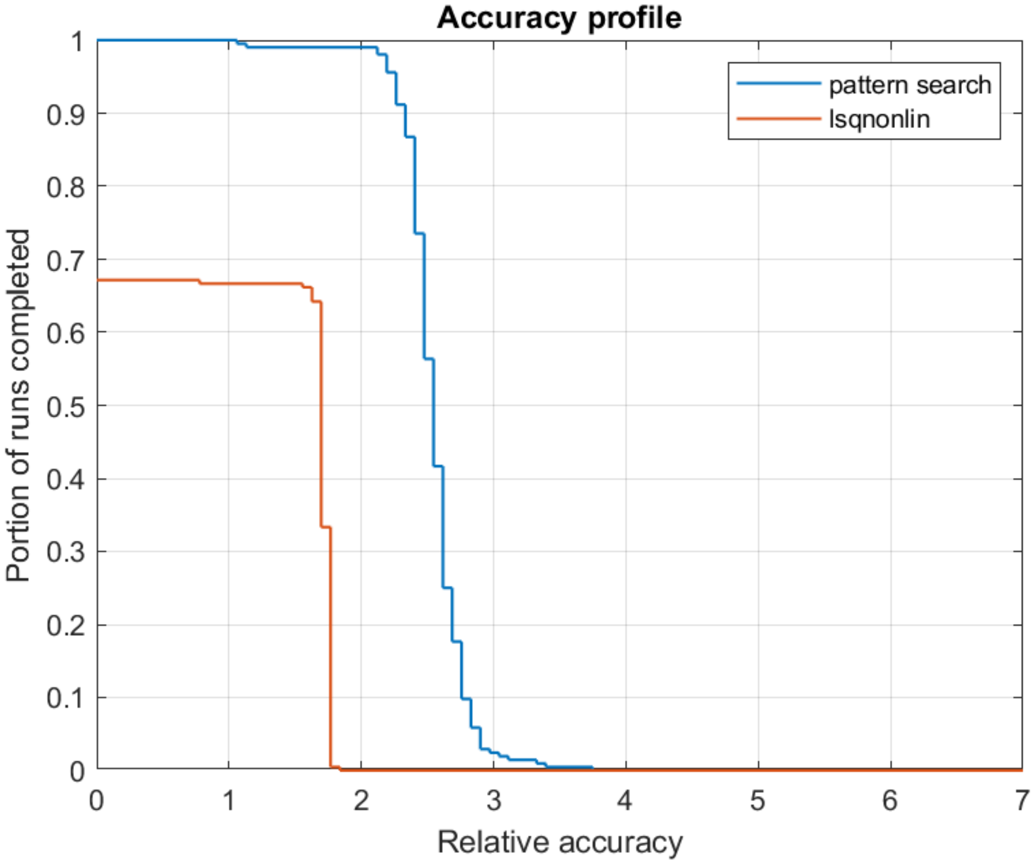


Figure 23 Accuracy profile comparison of patternsearch and lsqnonlin with noise

The presence of noise was extremely impactful to the accuracy of the estimation with the non-linear least square algorithm. Its solution went from being the one with the best RMSE result when compared to the patternsearch algorithm to performing worse than the patternsearch even in the accuracy aspect. The accuracy profile shifted from the baseline test of around four units, which means that the RMSE of the solutions increased on average by a factor of 10,000.

3.4.3 Test case with parasitic component in the model and without noise

In a real-life scenario, it is impossible to build a model that will perfectly match the characteristics of the real system. To replicate the same scenario, the algorithms were tested in a case where the model that produces the "target" characteristic was different from the one used for the identification. Therefore, creating an unbridgeable gap that the algorithms will try to mitigate. For this test, the model depicted in *Figure 17* was used to generate the "target" characteristic while the model in *Figure 16* was used inside the identification loop. As for the previous test the transitory characteristic was obtained by injecting at the end of the capacitor a 5V impulse. The parameters of the model to identify are presented in *Table 5*:

R_1 *[Ω]*	R_2 *[Ω]*	R_3 *[Ω]*	*L [nH]*	*C [nF]*
10	0.3	0.3	100	1

Table 5 Model parameters

Figure 26 presents the differences between the "target" characteristics obtained through the simulation of the model depicted in *Figure 17*, and the mean identification found by the patternsearch algorithm.

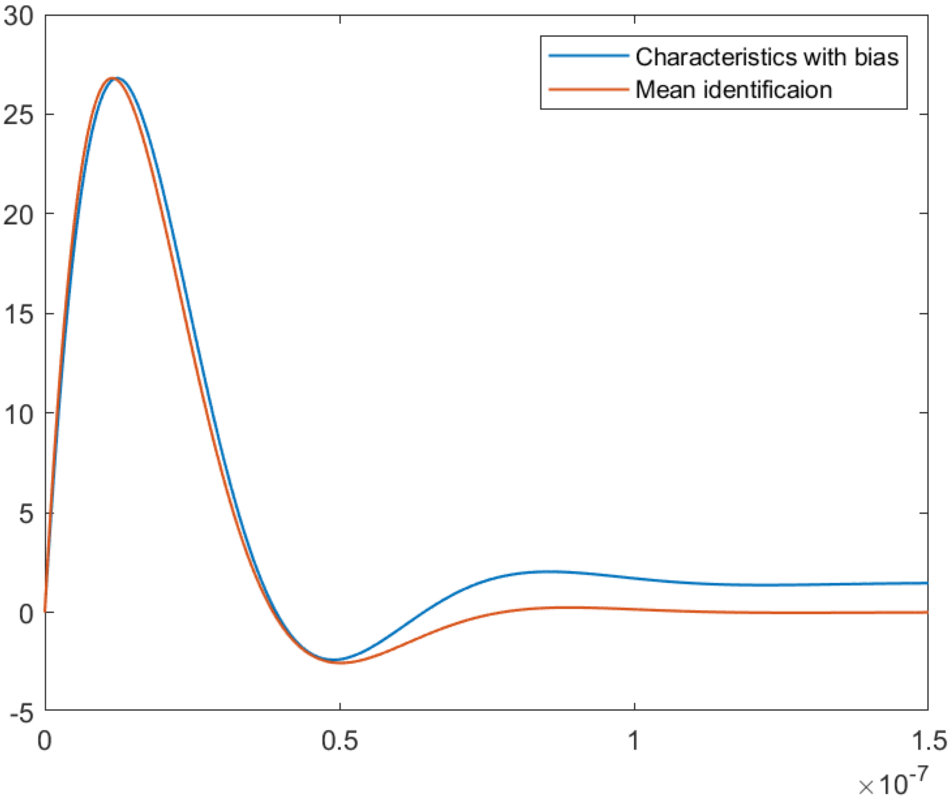


Figure 24 Comparison between the input and the identified characteristic

In order to study the quality of the estimation, a normal distribution was fitted to the frequency distribution of the parameters found. The parameters that were identified with the patternsearch algorithm were then organized and presented in the frequency distribution histograms of *Figure 27*.

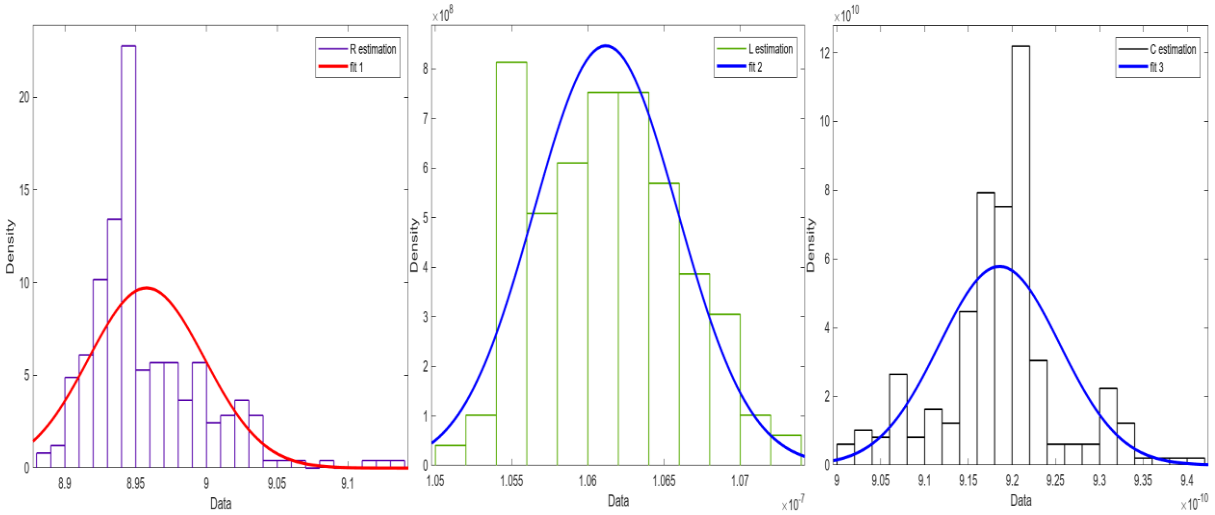


Figure 25 Distribution of the identified parameter: R (left), L (center) and C (right)

The mean and variance of those normal distributions fitted through the frequency distribution are reported in *Table 6*.

	R	***L***	***C***
Mean	8.95767	1.06118e-07	9.18573e-10
Variance	0.00168658	2.22169e-19	4.75453e-23

Table 6 Mean and variance of the normal distribution fitted to the estimated results.

By analyzing the parameter of the normal distribution that was fitted to the data, it is possible to notice that the optimizer compensated the bias in the model with a shift in the mean value of the identified parameter. The parameter that was the most affected by this compensation was the resistance, with a deviation from the baseline identification of roughly 11%. When comparing the estimated variance of this test with the one obtained in the baseline test but also in the previous case, where only the white noise perturbed the “target" characteristic, it is possible to notice that the order of magnitude of the estimated variance remains the same along the different tests, indicating that the presence of bias did not significantly alter the quality of the identification.

With the presence of bias between the two models, the threshold in the RMSE that used to classify between the problems that were correctly solved from the ones that were not was increased. From the tests, the RMSE of the solutions presented the same characteristics observed in the tests without bias that was presented in *Figure 19*, where most of the problems are solved with roughly the same RMSE while the remaining solutions present significantly higher RMSE. The tuning for the classification threshold was made by analyzing the RMSE of the results and choosing a value large enough to cut out any outliers but also not strict enough that acceptable solutions are discarded.

The data profile obtained in this test is presented in *Figure 28*. By analyzing the data profile of the patternsearch identification, it emerges that, for the threshold used, 99% of the problems were correctly solved. Similarly to the previous tests, most of the problems were solved within the three thousand function evaluation, after which there is a point of diminishing return, after which only a small portion of problems were solved.

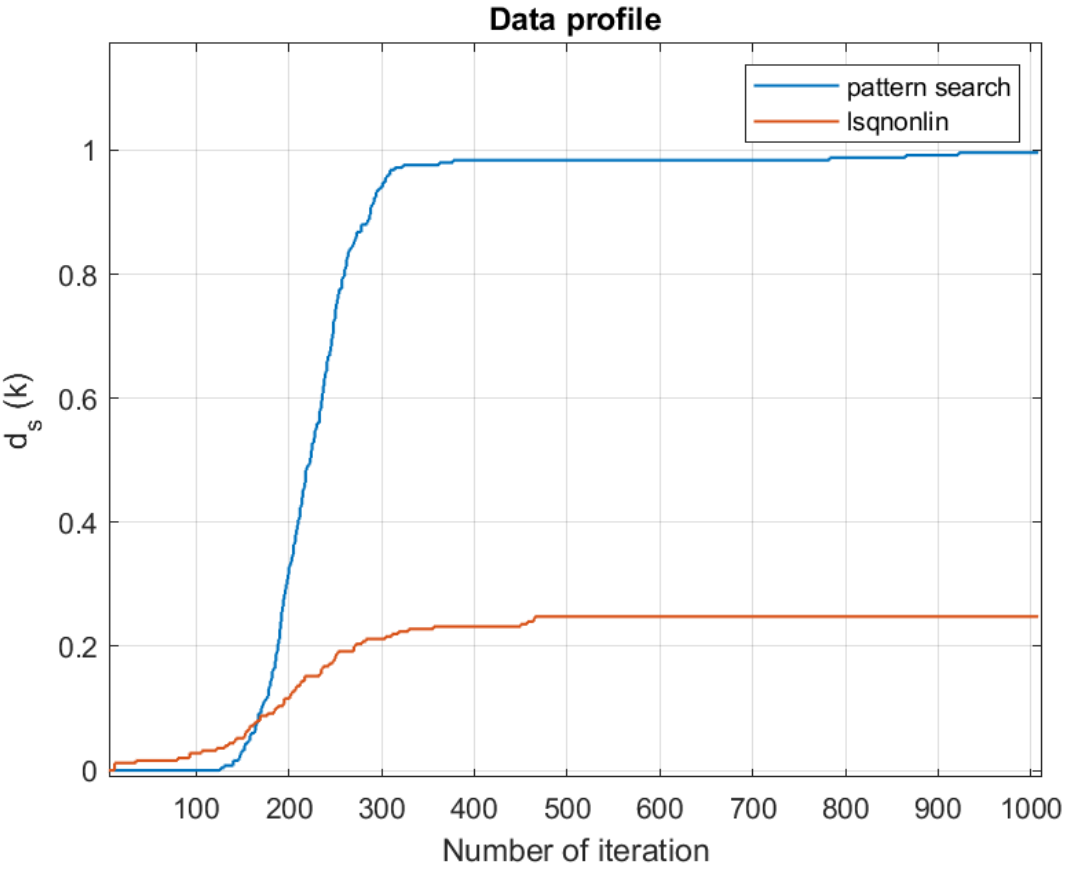


Figure 26 Data profile comparison of patternsearch and lsqnonlin with parasitic components

By analyzing the data profile of the nonlinear least square identification, it emerges that it performed poorly. At the end of the identification, only 26% of the test set was deemed solved, which is less than half of the problems solved in previous tests. The rate at which the problems were solved in relationship with the number of function evaluations was slightly better.

The accuracy profile obtained in this test is presented in *Figure 29*. By studying the accuracy profile, it is possible to notice that the curve of the *pattersearch* algorithm is characterized by a sharp fall, indicating that almost the entirety of the results of the identification has almost the exact same RMSE.

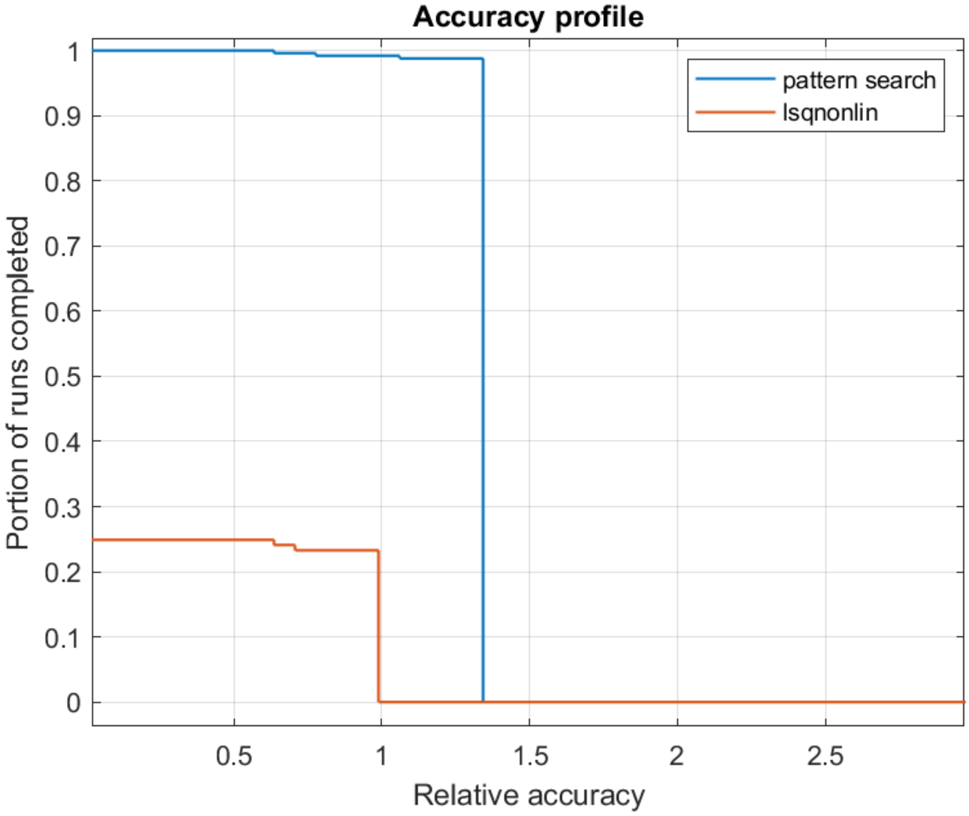


Figure 27 Accuracy profile comparison of patternsearch and lsqnonlin with parasitic component

Similarly, the accuracy profile of the lsqnonlin algorithm had the same shape, indicating that the presence of discrepancies between the models did not disrupt the consistency of the identifications. When comparing the accuracy profile of the two algorithms, it emerges that, even under these test conditions, the patternsearch algorithm could find results that, on average, have an RMSE that is roughly half of the RMSE of the results found with the classic nonlinear least square.

3.4.4 Test case with noise and bias in the model

To further resemble a realistic scenario, white noise was introduced to perturb the "target" characteristics. The ratio between signal and noise was 20db in order to replicate the same scenario of the previous test with noise. Other than the added noise the model did not present ulterior differences. The parameters used are the same as the ones used in the previous test and depicted in *Table 5*. As for the previous test the transitory characteristic was obtained by injecting at the end of the capacitor a 5V impulse. The difference between the inductor current transitory that is the input of the identification, "target", characteristics and its best approximation is presented in *Figure 30*.

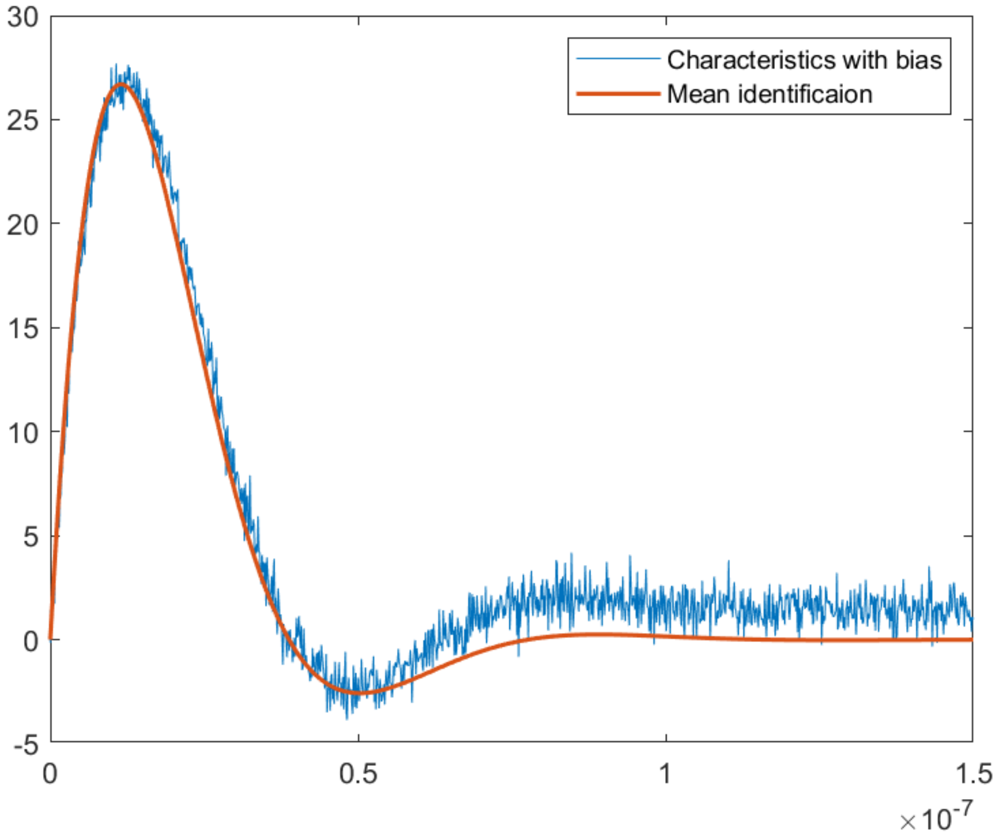


Figure 28 Comparison between the input and the identified characteristic

In order to study the quality of the estimation, a normal distribution was fitted to the frequency distribution of the parameters found. The parameters that were identified with the patternsearch algorithm were then organized and presented in the frequency distribution histograms of *Figure 31*.

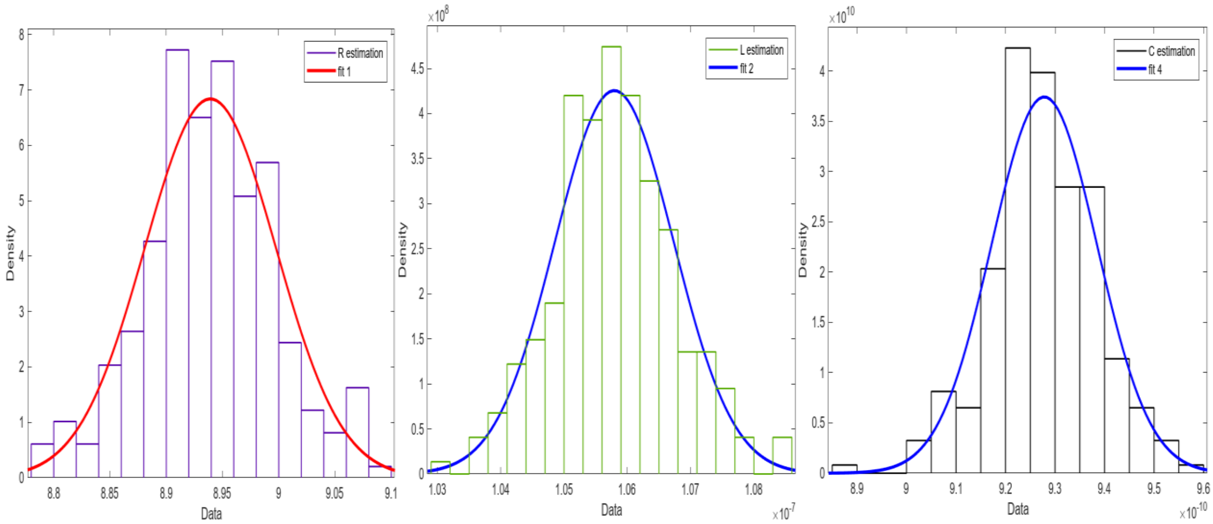


Figure 29 Distribution of the identified parameter: R (left), L (center) and C (right)

In order to assess the quality of the estimation, a normal distribution was fitted to the frequency distribution of the parameters found. The mean and variance of those normal distributions are reported in *Table 7*.

	R	***L***	***C***
Mean	8.93919	1.05794e-07	9.27824e-10
Variance	0.00340189	8.78708e-19	1.13698e-22

Table 7 Mean and variance of the normal distribution fitted to the estimated results.

Similarly to the previous test case without noise, the algorithm compensated for the bias in the model with a shift in the mean value of the identified parameter. The introduction of noise in the estimation did not interfere heavily with the quality of the estimation, as shown by the variance of the normal distribution that was fitted on the data that has the same order of magnitude as the previous test without noise. Furthermore, the mean values of the parameter estimated did not suffer from any significant deviation with respect to the previous case without noise.

As shown in the data profile presented in *Figure 32*, the presence of noise improved the total portion of problems that were solved by the non-linear least square. The number of function evaluations required by the non-linear least square to solve a problem was, on average, higher than the number of function evaluations required by the patternsearch algorithm.

By analyzing the data profile, it is remarkable to notice that the patternsearch algorithm did not show any meaningful degradation in its efficiency to converge to a solution, proving its robustness. 90% of the problems were solved within the three thousand function evaluation mark. At the end of the process, the patternsearch algorithm was able to solve 249 problems out of the 250 of the test set.

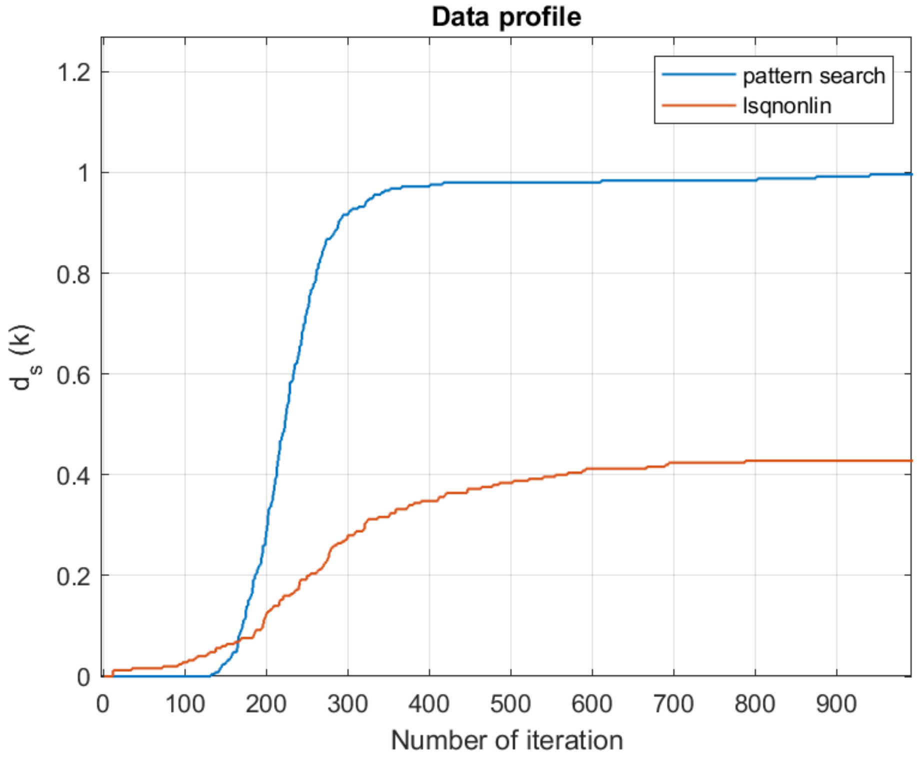


Figure 30 Data profile comparison of patternsearch and lsqnonlin with noise and parasitic component

Lastly, the accuracy profile presented in *Figure 33* did not show any significant difference from the previous case. Both algorithms produced results with consistent accuracy, but the patternsearch algorithm produced solutions that, on average, have an RMSE that is roughly half of the RMSE of the results found with the classic nonlinear least square.

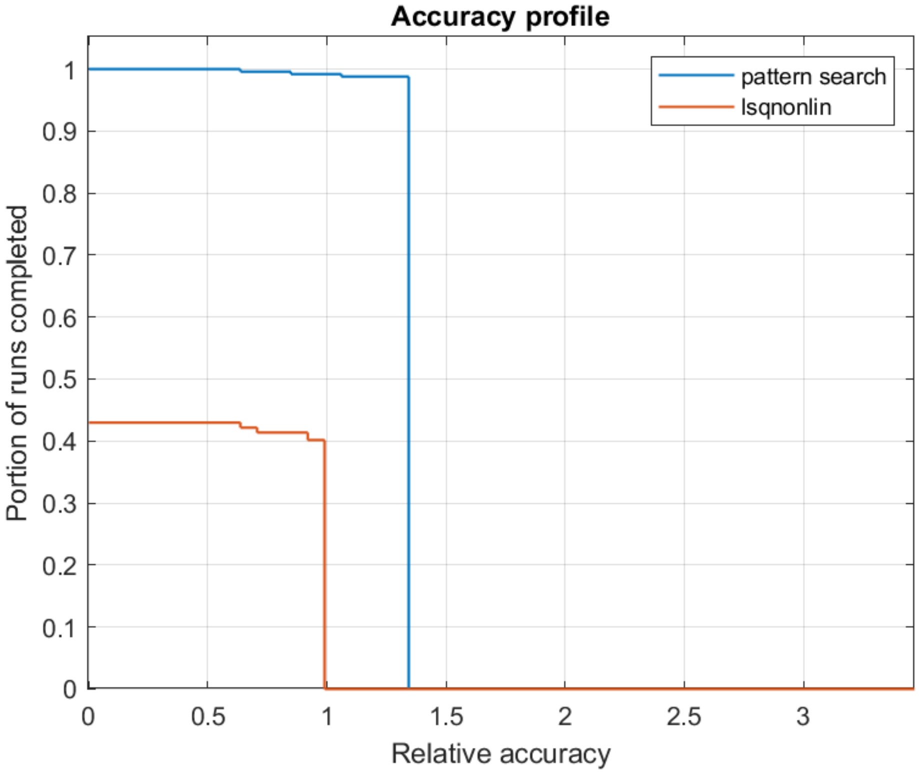


Figure 31 Accuracy profile comparison of patternsearch and lsqnonlin with noise and parasitic component

3.5 Comparison with other optimizers

In the literature, other optimizers were used in order to solve similar problems as in [9] and [8]. In those papers, the algorithms proposed were the particle swarm optimization in [9] and the Bayesian optimization in [8]. Since those optimizers proved their effectiveness in similar applications, some tests were performed in order to compare their performances with the patternsearch algorithm.

The tests were performed in the simplest case with neither noise nor bias and with a cost function defined in the same way. The set P of problems comprised 100 initial condition perturbations, and all optimizers solved the same set P. Even if the dimension of P was reduced, it was nonetheless sufficient to highlight the main characteristics of those optimizers.

From the accuracy profile in *Figure 34*, it is possible to notice that the "patternsearch" and the "PSO" algorithms obtained similar results in terms of accuracy. "Bayesopt", on the contrary, performed the worst. It has to be stated that the accuracy of the "bayesopt" algorithm was hindered by the time constraints imposed on the optimization process. The choice of implementing a time limit for this type of optimizer was a necessary countermeasure in order to prevent the optimizer from stalling since the algorithm used did not have the option to exit the optimization once the improvement on the cost function was below a defined tolerance. Nevertheless, the chosen time limit was three minutes, which was more than three times the time required for the other algorithm to solve similar problems.

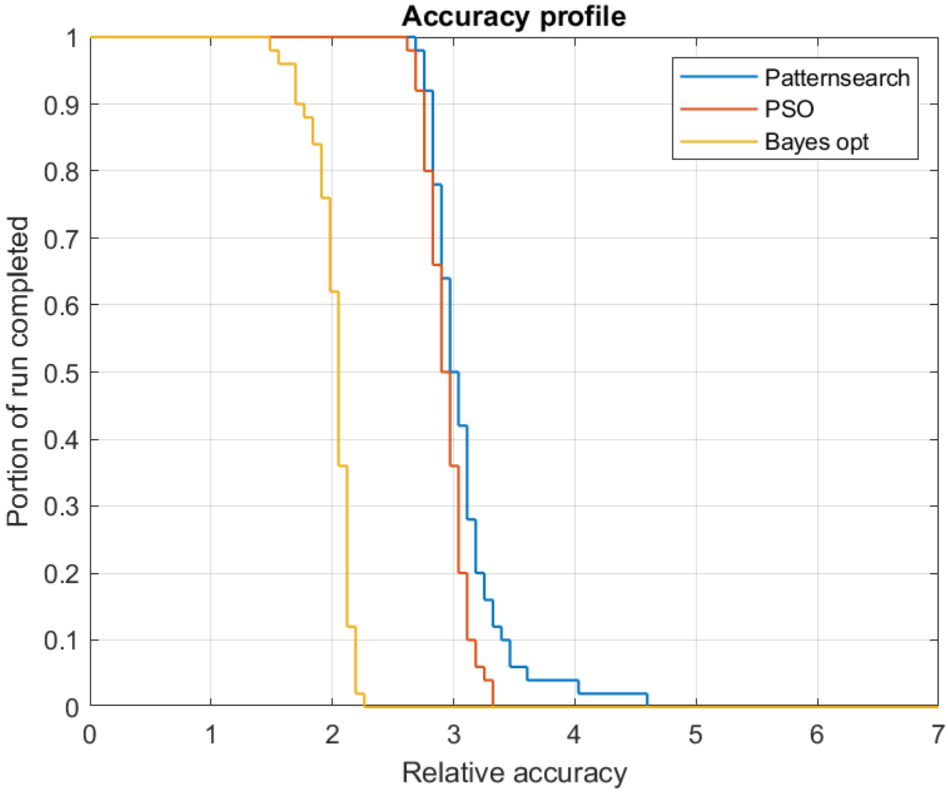


Figure 32 Accuracy profile of three different optimizers

When inspecting the data profile, it is possible to notice that even if the particle swarm optimization was able to find a solution with accuracy comparable to patternsearch, the number of function evaluations required was extremely high. The bayesopt algorithm, as discussed in [8], requires less iteration than the particle swarm optimization but still more than the one needed for the patternsearch.

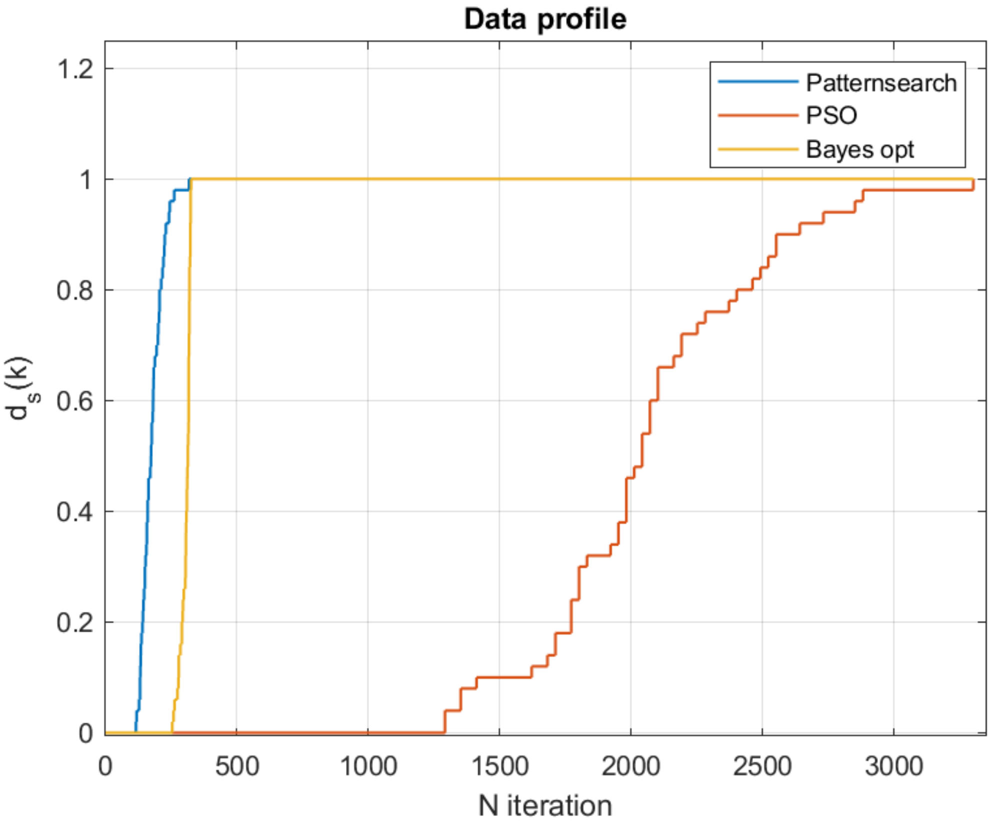


Figure 33 Data profile of three different optimizers

From the test performed, it emerges that the patternsearch algorithm is the most suitable for the problem under exam since it provided the best results in both the accuracy of the solution and efficiency in completing the optimization.

Chapter 4

Application to a Buck converter

4.1 Identification workflow

With the simple RLC circuit, it was possible to perform a large number of iterations to assess the performance of the solution proposed for the identification. The next step was to implement the same strategies that proved to be effective in a simple test circuit to a more complex electronic device. The buck converter was the electronic circuit chosen to test the proposed identification approach to prove its effectiveness in a more realistic scenario. As a reference for the main parameters of the model, it was chosen the buck converter model TPS40200EVM-002, since was also studied in the paper [1].

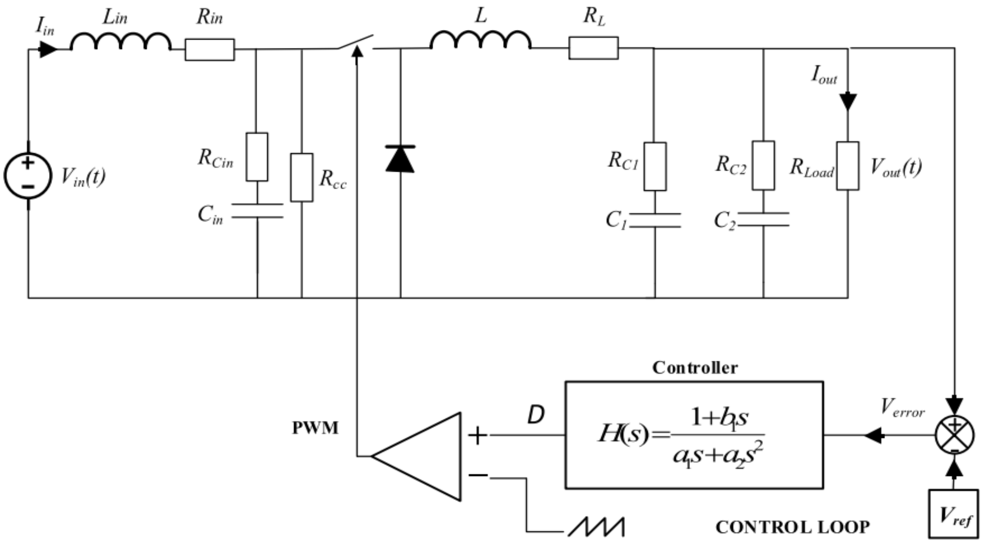


Figure 34 Buck converter studied in paper [1]

It was possible to implement an accurate model of this device since Texas Instrument, the manufacturer of this design, provided in the datasheet of the device all the relevant parameters.

The nominal values of the main parameters of this circuit are presented in *Table 8*:

Electrical values

L	33µH
C_1	20µF
C_2	440µF
R_L	60mΩ
R_{C1}	60mΩ
R_{C2}	65mΩ

Controller values

a_0	0
a_1	4.7×10^{-4}
a_2	1.6×10^{-9}
b_0	1
b_1	4.7×10^{-4}
b_2	0

Table 8 Nominal values of the main component(left) and of the controller (right)

As in the previous scenario, the identification process was made by measuring a "target" characteristic and by minimizing the difference between observed and simulated responses. For the buck converter, the "target" characteristic used for the identification is the ripple in steady state condition. To accurately measure the output ripple required for the identification, high-frequency sensors are needed. Since, ideally, the final application of this identification process is to have an embedded system that monitors the health of the circuit, making this system a financially valuable solution is a requirement. Thus, the number of sensors was reduced compared to [1] and only one voltage sensor was used to measure the voltage characteristic of the converter's output. This design choice will inevitably hinder the accuracy of the identification. This compromise was made since the system's objective was not to obtain the most accurate identification but to achieve a level of accuracy high enough to detect the degradation of the components.

Due to the system's complexity, multiple parameter combinations could lead to a characteristic similar to the "target" one. This results in a cost function that presents a non-neglectable number of local minima. The risk for the algorithm to end in a local minimum is high enough to discard the option of a single iteration as a sufficiently accurate option for the identification. This is especially true for the MOSFET which presents itself as the most challenging component when it comes to identifying its health due to the small deviation in its failure indicator. The failure indicator used for the MOSFET was the ΔR_{on} , from the research on the evolution of this parameter, is presented in [33], it was determined that a $\Delta R_{on} = 50m\Omega$ was a safe threshold to determine the end of life of the component under analysis.

To compensate for the inaccuracy of the identifications the process the following process was implemented. For each "target" characteristic, ten identifications were performed with different initial conditions. The initial conditions were generated by perturbing with a white noise the previously identified parameter or, in the case of the first iteration, the nominal value of the parameter. The final result of the identification was obtained by mediating the different identification results. The workflow presented is presented in the graph in *Figure 37.*

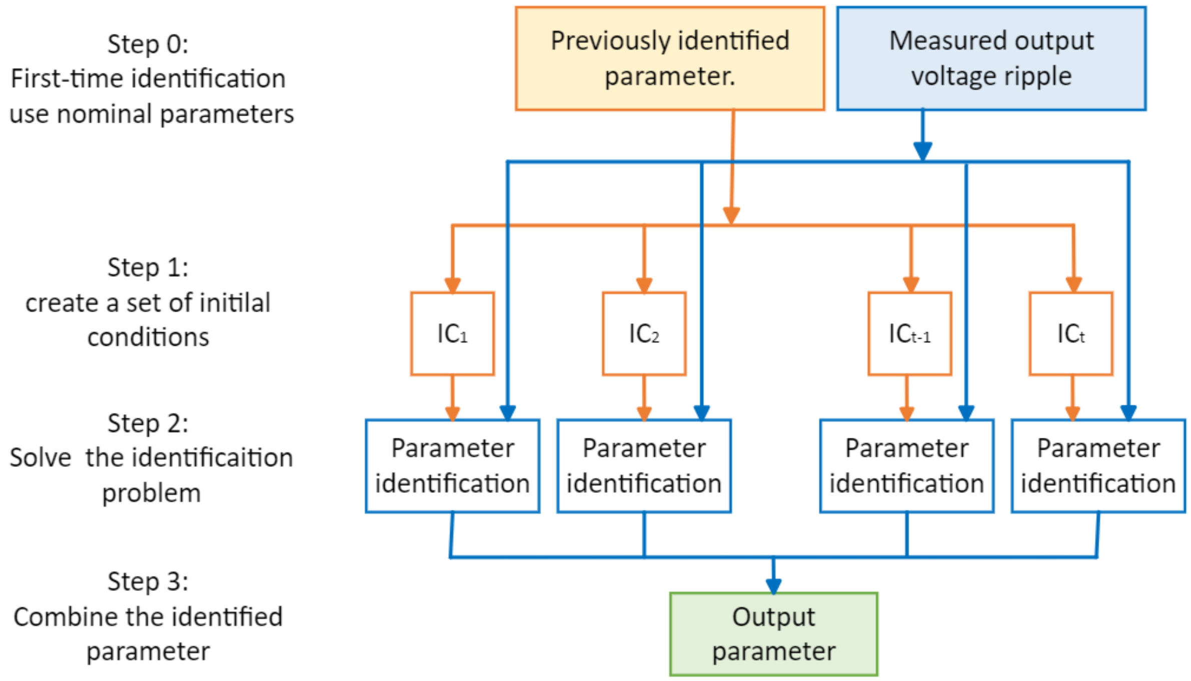


Figure 35 Data process cycle

The workflow presented was heavily influenced by the work of the paper [34] where the previously obtained parameters are implemented as initial conditions for the next identification and from the ensemble methods presented in [35].

It must be noted that this approach is sensitive to the frequency of the measurement taken. Reducing the time gaps between the measurement will inevitably bound the deviation of the parameter to more limited values.

Smaller deviations between the parameter to identify and the initial condition of the optimizer will inevitably cascade into a reduction in the number of function evaluations required for the parameter estimation and therefore time. Increasing the time efficiency of the optimization is therefore a crucial point to improve the performance of the system. One advantage of this workflow is that limitations in hardware capability can be avoided not only by upgrading the hardware but also by splitting the computational burden to multiple devices.

4.2 Generation of the “target”

The “target” characteristic that was used for the identification was generated with the Simulink model shown in *Figure 38*.

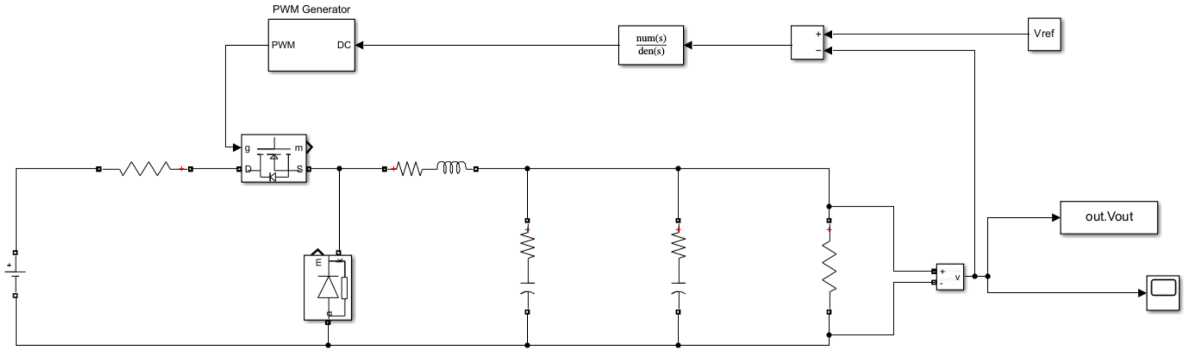
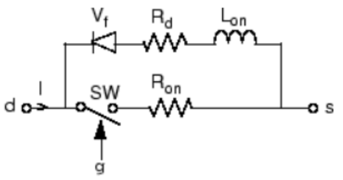


Figure 36 Simulink model used to generate the "target" characteristic.

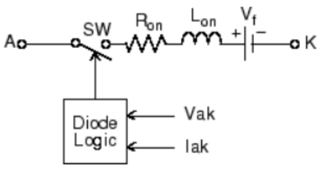
Since Simulink has already built in model for the MOSFET and diode, the schematic slightly differs from the one shown in *Figure 36*. From the TPS40200EVM-002 BOM [36], it was possible to find the MOSFET and diode used for the converter which are FDC5614P and MBRS360, respectively. Nominal parameters of those components were extracted from their respective datasheets. Model implemented by Simulink and respective values of the parameter for the FDC5614P transistor were taken from [37] and are presented in *Figure 39*:



	Parameter
R_d	105mΩ
R_{on}	0.135Ω
V_f	−0.8V
L_{on}	0μH

Figure 37 The MOFET model implemented by Simulink(left) and its parameter (right)

Model implemented by Simulink and respective values of the parameter for the MBRS360 diode were taken from [38] and are presented in *Figure 40*:



	Parameter
R_{on}	0.2Ω
V_f	0.63V
L_{on}	0μH

Figure 38 The diode model implemented by Simulink(left) and its parameter (right)

For the model that traces the target characteristic, it was chosen to adopt a higher resolution than the one used inside the estimation loop. This choice was made not only to reduce the computational burden of the model inside the loop but also to replicate the unbridgeable gap that is unavoidable when a mathematical model is used to replicate physical behavior.

The solver implemented for this model was the ode23tb variable-step continuous since it is suitable for solving nonlinear stiff problems [1]. In this problem, it has faster convergence and higher accuracy than other solvers, such as the ode45 and ode15s solvers [1]. The max step for the solver was set to 2×10^{-10} seconds, in order to replicate the 5GHz sampling frequency used in [1], and the refining factor was 10. Increasing the refining factor was necessary not only to improve the output but also to avoid consecutive zero-crossing violations.

4.3 MATLAB model implementation

For the identification, a second model was assembled. The requirements for this second model were different from before since it is required to run the model iteratively. Therefore, the simulation time becomes one of the most relevant factors in the design of this model. Any inefficiency in the set-up of the model used for the optimization process will inevitably build up to a considerable time increment at the end of the identification. The time increment at the end of the optimization ranged from a couple of minutes to an entire hour on the test performed.

With the objective of optimizing the simulation time, different parameters of the set-up were tuned. The max step was increased to 5×10^{-3} seconds, the refining factor was tuned down to 5, and the simulation optimizer was implemented in order to verify that all the settings were optimal for the problem in exam [39].

Optimizing the set-up of the simulation was the first step. The changes implemented were not limited to the setting of the simulation. The model was simplified by removing the dynamic control of the duty cycle. The Simulink model that resulted from this process is presented in *Figure 41*.

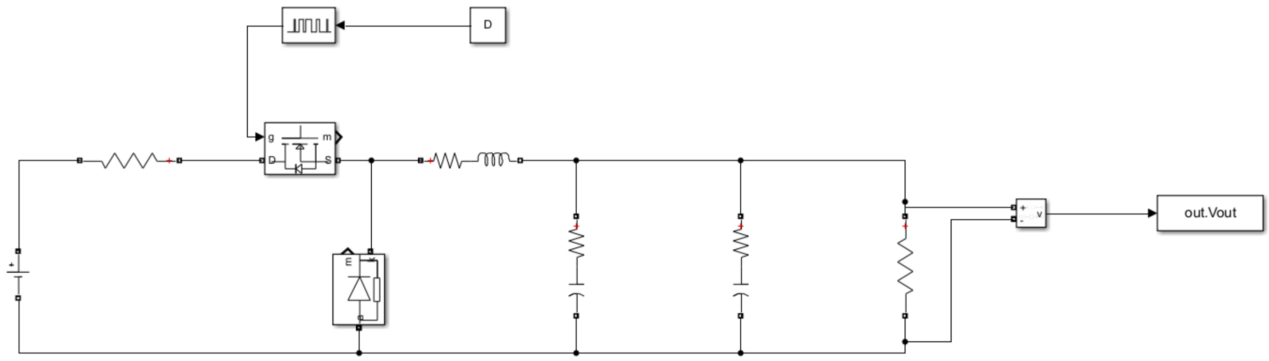


Figure 39 Model used to generate the characteristic.

All the changes implemented were aimed at minimizing the computational burden of the model complexity and, therefore, its simulation time. This change led to a significant reduction in the simulation time. The single function evaluation with the same setting of the solver went from around 9.8 seconds for the initial tests to 2.4 seconds for the last model design. It has to be stated that the simulation time is also limited by the capabilities of the hardware, and therefore, the absolute values presented must be taken with a grain of salt. The detailed design choices that allowed to shave more than half of the simulation time are discussed in the following paragraphs.

4.3.1 Duty cycle identification

Removing the dynamic control of the duty cycle was the most impactful design choice that was made both in terms of the workarounds required and the benefit that this design choice yielded. Using an open loop configuration removed the negative impact that the simulation errors had on the feedback but also removed the dynamic controller in a continuous time domain, which was computationally expensive. With looser requirements on the simulation, it was possible to produce a steady-state output that was comparable with the one observed with the original model with a max step for the time domain of the solver that was one order of magnitude larger.

Without a controller that dynamically computes the duty cycle, it is necessary to define one that will lead to the desired steady-state condition.

Different approaches could be implemented in order to define the duty cycle. One of those solutions could be to analytically compute a new duty cycle for each iteration of the optimization process in order to obtain the desired mean voltage value.

This could be a viable option since all the variables are known. The mean voltage of the ripple could be extracted from the measured "target" characteristic, and the values of the components are known since they are the parameter chosen by the optimizer in that iteration.

This approach would allow to introduce to the cost function an error contribution that would take into account the difference between the duty cycle of the "target" and the identified characteristics.

From the test performed with this approach, there were no measurable improvements in the convergence efficiency or accuracy of the optimization. The lack of any significant improvements led this approach to be discarded in favor of a more straightforward approach that helped to streamline the code and remove a possible source of miscalculation.

The solution adopted was to use the duty cycle estimated from the "target" characteristics and then use the same value for all the iterations of the model. In this way, the duty cycle is computed only once outside the function that will be reiterated. This approach did not introduce any additional computational burden to the cost function, therefore optimizing the function evaluation time. The duty cycle estimation was made by locating the discontinuities in the ripple caused by the MOSFET switching on and off. The side-to-side comparison of the output voltage ripple and the MOSFET duty cycle is highlighted in *Figure 42*.

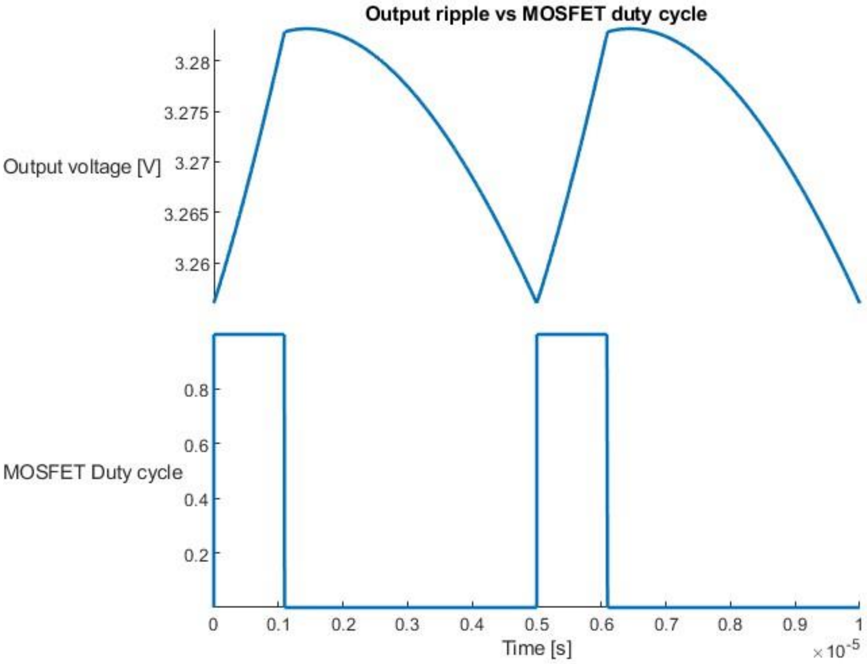


Figure 40 Output voltage ripple side-to-side the MOSFET duty cycle

Since the estimation of the duty cycle is a one-off computation, there was not any particular concern regarding its computational efficiency. Therefore, the discontinuity was found by implementing the built-in MATLAB function "ischange". The estimation of the duty cycle was made on a single ripple period. This was a feasible solution because there was no noise to disturb the characteristics. In a real scenario, in order to obtain a higher quality estimation of the duty cycle and not hinder the quality of the parameter estimation, it would be preferable to evaluate the duty cycle and "target" characteristics over multiple ripple periods.

4.3.2 Initial condition

Introducing a static duty cycle reduced the computational burden but inevitably increased the time required to achieve steady-state conditions, consequently increasing the required simulated time.

In order to mitigate this effect, the initial conditions of the active components were initialized as the mean value of the steady-state operating point. The initial condition for the capacitor was set as the mean voltage value of the “target” ripple. The initial condition of the inductance was the mean value of the current from the model with the minimum cost function to that iteration. Initializing the active components in this way allowed to drastically reduce the time required for the system to achieve a steady state, as shown in *Figure 43*.

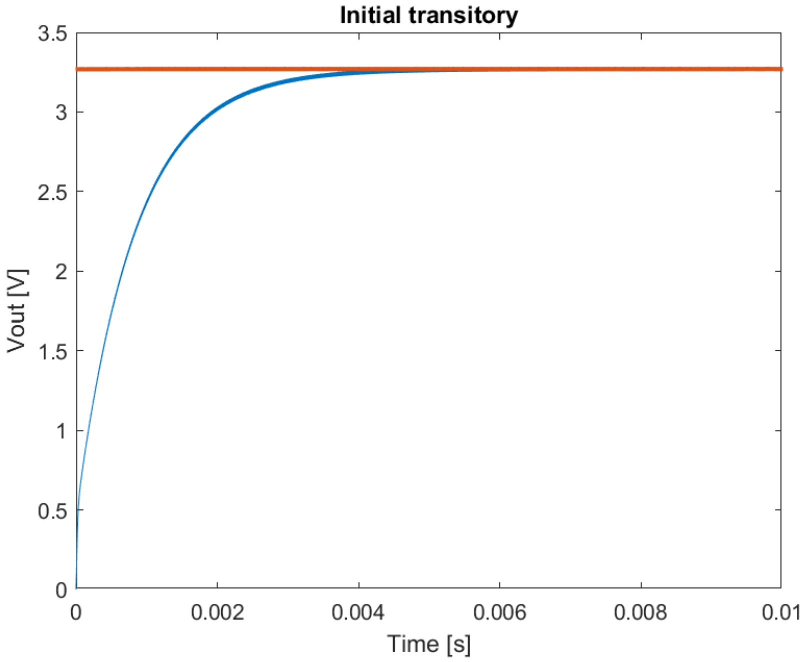


Figure 41 Time response with initial condition to zero(blue) and different from zero (orange)

In the test performed, in order to achieve the same identification results, the simulated time with all the initial conditions set to zero required a simulated time of 0.008 s. In the case of initial conditions set as discussed before, the required simulated time was 0.001 s. In order to ensure the lack of any residual transient in the final version of the program, the simulated time was set to 0.002s, doubling the minimum requirements for this approach.

4.4 Signal processing

There are two signals that need to be processed in order to compute the cost function: the "target" characteristic and the simulated one. The first step is to filter the "target" characteristic from measurement noise. Once that is done, it is required to align and finally synchronize the data samples of the simulated characteristics with the "target" ones. A representation of the process is presented in *Figure 44*.

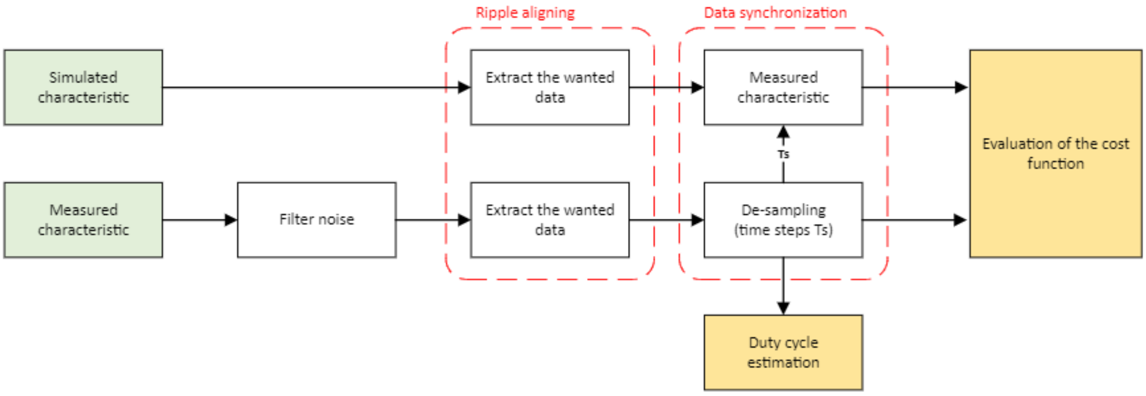


Figure 42 Signal processing

The previous figure highlighted that there are two main steps required in order to be able to evaluate the cost function: Ripple aligning and data synchronization.
It has to be noted that the processing of the measured and simulated characteristics is not performed in parallel due to the iterative nature of the approach. The measured data was processed only once before starting the optimization process.

4.4.1 Ripple aligning

In order to compare the two curves, a reference point that is ubiquitous in all ripple characteristics must be chosen. By analyzing the ripple, it is possible to notice that it has a very sharp local minimum that is easy to detect, and it is always present irrespectively of the parameters of the model, since it is caused by the working principle of the device. For those reasons, the local minimum was chosen as the reference point from which the ripple will be cut.

In the case of the simulated characteristic of the model, it is necessary to wait for the device to reach steady-state condition before sampling the ripple that would be compared to the "target" characteristic. In order to avoid any residual transitory components that could affect the quality of the identification, only one ripple period was used to generate the cost function.

4.4.2 Data synchronization

In this test case, the model used for the data generation and the one used for the identification is characterized by different options regarding the max step of the simulation. Therefore, it will be necessary to synchronize the data to the same time step. This step is not exclusive to the test case under analysis but also to the real-world scenario, where the fixed measured sampling step will differ from the step used by the solver to simulate the model.

The first step for the data synchronization was to de-sampling the “target” ripple characteristics. The “target” characteristic was produced with an extremely high resolution with the objective of not only producing a curve with the highest quality possible but also mimicking the sampling rate of 5 GHz implemented in [1]. The number of points considered for one period of the ripple was reduced from more than 5000 to 2000. Each of the points of the de-sampled set is separated with a fixed step Tp. The aligning and the de-sampling of the “target” characteristic are performed before entering the optimization process in order to avoid reiterating a process that will not change for a fixed “target” characteristic.

The second and last step of the data synchronization is to ensure that for each timestamp value that is present in the de-sampled “target” characteristic, there must be a corresponding value of the simulated characteristic. Since the “target” curve is characterized by a fixed step Ts and the simulated one, for efficiency purposes, was created with a variable step, an approximation was required. Due to the high density of points, it is possible to approach this problem with a 1D nearest neighbor by approximating the value with the desired timestamp by taking its nearest function evaluation. This crude approach was used initially due to its ease of implementation and, from the test performed, did not impair the ability to converge. Therefore, a linear interpolation was implemented with the objective of refining the script and minimizing any sources of approximation error with minimal increase in the computational burden.

$$v(kT_p) = \ kT_s - t_n \ \frac{v\ t_{n+1} \ - v\ t_n}{t_{n+1} - t_n} + v\ t_n \quad with\ k \in \mathbb{N}\ 0 \leq k \leq 2000 \tag{4.1}$$

Where t_{n+1} and t_n are the time stamps of the variable step dataset between which the time stamp kT_s is included.

4.5 Error function

The cost function used for the identification was constructed on a single period of the output voltage ripple. This was a design choice made in order to minimize the following sources of error:

- **Residual transitory** Even if minimal across several periods a small residual transitory could be present and negatively impact the quality of the identification.
- **Duty cycle variation** The continuous controller implemented by the device to stabilize the output voltage tries to compensate for the ripple by modifying the duty cycle. This minimal fluctuation in the controlled variable impacts the steady state ripple by introducing a small periodic contribution with a period several times larger than the ripple ones. An approach that utilizes multiple periods to construct the objective function would require to: Estimate the duty cycle for each ripple period and then aligning the target and the estimated characteristic, to have the duty cycle of the two characteristics to coincide.

Once the section of the ripple that will be used for the identification of a cost function was defined and all the data processing required was finished, it was time to define the objective function. As in [1], the voltage was standardized with respect to the "target" as described in the *Equation 4.2*.

$$v\ t\ = \frac{v\ t\ - v_{mean_t}}{\sigma_t} \tag{4.2}$$

Where $v\ t$ is the output voltage, v_{mean_t} is the mean value of the "target" ripple and σ_t the standard deviation of the "target" ripple. In order to simplify the tuning of the f_{cp}, it was required to normalize not only the voltage but also the time of the characteristics in exam. The time was normalized with respect to the ripple period T_p as follows.

$$t = \frac{t}{T_p} \tag{4.3}$$

Once data were normalized, the next step was to construct the objective function. To define the f_{cp} component to the objective function, a procedure similar to the one applied for the identification of the RLC was implemented.

The procedure implemented to choose the characteristic points to construct the f_{cp} was the following. First, the optimization process was run using only the f_{RSS}, and its convergence behaviour was studied. From those tests, it emerged that the two main characteristics that the optimizer needed to correct were the shape of the transient when the MOSFET was in the off position and an offset between the two characteristics. With the objective of prioritizing the convergence of those errors, two points were used to build the fcp.

The first contribution added to the f_{cp} was the timing and voltage difference of the peak. This contribution was chosen with the objective of helping the algorithm to converge to the correct shape of the transitory. This contribution for a set of parameters x was defined as follows.

$$f_{peak}\ x\ \ = \frac{1}{2}\left[\left(t_p^* - t_p\right)^2 + \left(\ v^*\left(t_p^*\right) - v\left(t_p\right)\right)^2\right] \tag{4.4}$$

Where t_p and t_p are the normalized timing of peak in the "target" and in the simulated characteristic, $v^*\left(t_p^*\right)$ and $v\left(t_p\right)$ are the standardized voltages with respect to the peak of the "target" characteristics. Since both the time and voltage of the characteristic were standardized and had similar values, no coefficients were defined when mixing together the two contributions.

The third contribution that was added to the cost function was the voltage difference of the ripple endpoint. This contribution was chosen since it proved to be an effective aid for the algorithm in removing the eventual offset between the two characteristics. This contribution for a set of parameters x was defined as follows.

$$f_{end}\ x\ \ = [v^*\ 1\ \ - v_t^*\ 1\ ]^2 \tag{4.5}$$

All those contributions were used to define the cost function that the optimizer will minimize. The cost for a set of parameters x was defined in the *equation 4.6.*

$$f_{Cost}\ x\ \ = \alpha f_{RSS}\ x\ \ + \beta f_{peak}\ x\ \ + \gamma f_{end}\ x \tag{4.6}$$

Where:

$$\alpha + \beta + \gamma = 1\ \mathit{with}\ \ \alpha, \beta, \gamma \in \mathbb{R} \tag{4.7}$$

As in the simplified example with the RLC circuit, the tuning of the different contributions was made by defining a set of problems that were fed to the algorithm implementing different weights for the cost function values. After a defined number of functions evaluations, the algorithm was stopped, and its progress assessed.

The process used to tune the cost function was similar to the one that was implemented for the simpler circuit and proved to be very effective. To start, an identification was made by implementing solely f_{RSS}, this served as a reference point in order to assess the improvements that were made by tuning the cost function. This initial test highlighted the computational expense required to perform this identification; it took in fact around 19 hours to complete this initial identification. Waiting for the algorithm to finish the estimation was not a feasible option since especially in the first tuning phase of the algorithm the optimizer was prone to be stuck for multiple function evaluation in a local minimum. It was chosen to evaluate the progress in the estimation after a fixed amount of function evaluation, since between the ones described in [31] it was the method that was more in line with the requirement of the case. The other contributions of the cost function were gradually increased once the improvement in the identification started to reach the point of diminishing return, the cost function was then considered sufficiently tuned. Further tuning was unnecessary since this was a proof of concept and specializing the algorithm to solve a specific problem could backfire when the parameters of the system will drift from the nominal ones.

As previously noted, to improve the quality of the identification result, it was chosen to repeat multiple times the identification problem with perturbed initial conditions. This initial set of problems was generated by disturbing the a priori knowledge, that comes in the form of the nominal values, with a white noise. This method proved to be an effective way to generate a set of results, that can afterwards be analyzed through statistical methods. For this application, it was chosen to simply mediate the different results of the multiple identification.

This is the point in the workflow where the concept of digital twin was implemented. Instead of discarding the previously identified parameter and starting the next identification from the nominal values, the previously identified parameters were implemented to generate the set of initial conditions for the next iteration of the process. A similar recursive process was implemented in [34]. This is a feasible solution since in this study the focus on the identification of physical parameters that from definition will evolve in a continuous manner but are also fault indicators that generally drift in their values with years of normal use. This last property allows to considering this type of method that requires multiple hours to finish a valuable option.

Implementing the previously identified parameters to generate the next set of initial conditions allowed to optimize the time required for the process of identification. Since it allowed to reduce the gap between the initial conditions and the final result.

To obtain the result that will be discussed in the following paragraph it was decided to use a set of 10 initial conditions, generated with a normal distribution with variance $\sigma^2 = 2.7\overline{7} \times 10^{-4}$ and mean $\mu = 0$. The variance was calculated to have $3\sigma = 0.05$ meaning that 99.7% of the initial condition generated did not exceed a 5% deviation from the result from which they were derived.

4.6 Identification results

To extract meaningful data from the test performed it is crucial to select identification problems that may highlight any possible shortcomings of the method used. To study the basic behavior of the system it was chosen to proceed as follows. First, choose one of the parameters that were highlighted as a failure indicator of a component, then generate a set of identification problems where the chosen parameter is steadily increased to mimic the degradation of the associated component. The tests performed were designed to test two main factors: how much the method is able to pinpoint the root cause of the changes in the steady state characteristic and eventually quantify how much the deviation of one parameter would influence the identification of the others and test the accuracy capabilities of the algorithm while following the drift of the parameter.

As stated in [7] the expected variation in the ESR before it reaches the breaking point, in the case of an electrolytic capacitor, can be very significant and reach peaks of 2.78 times the initial value. Having such a noticeable increase in its failure indicator makes one of the two most critical components also the easiest to diagnose for issues and the best point to start testing the capability of the algorithm.

4.6.1 Capacitor identification

With the objective of simulating a gradual increase of the ESR of one capacitor over time the following three ripple characteristics were simulated: The first ripple that was simulated was the baseline where all the parameters were the nominal ones, the second and third present an increased ESR of 5% and 10% respectively. The smallest ESR present in the model was chosen, in order to further test the capabilities of the algorithm. The array of the parameters used in the tests is presented in *Table 9.*

	θ_0	θ_1	θ_2
L	33µH	33µH	33µH
C_1	20µF	20µF	20µF
C_2	440µF	440µF	440µF
R_L	60mΩ	60mΩ	60mΩ
R_{C1}	60mΩ	60mΩ + 0.3mΩ	60mΩ + 0.6mΩ
R_{C2}	300mΩ	300mΩ	300mΩ
R_{on}	0.135Ω	0.135Ω	0.135Ω

Table 9 vector of parameters used in the test

The graph presented to depict the data is structured in the following way: The points on the x-axis are meant to represent a point in time where the parameters are set, and the y-axis depicts the normalized deviation of the parameters to the respective nominal values. $x(\theta_1)$ is the first identification made by the algorithm, where R_{c1} presented a 5% deviation from the nominal value. Lastly $x(\theta_2)$ is the second identification where R_{c1} presented a 10% deviation from the nominal value. Instead of having a cluster of points that spread along the y-axis it was decided to present the data using a box chart. or box plot, provides a visual representation of summary statistics for a data sample. Given numeric data, the corresponding box chart displays the following information: the median, the lower and upper quartiles, any outliers (computed using the interquartile range), and the minimum and maximum values that are not outliers [40].

The graph presented in *Figure 45* represents the result of the identification for the parameter Rc1, the ESR parameter that changed between the identifications.

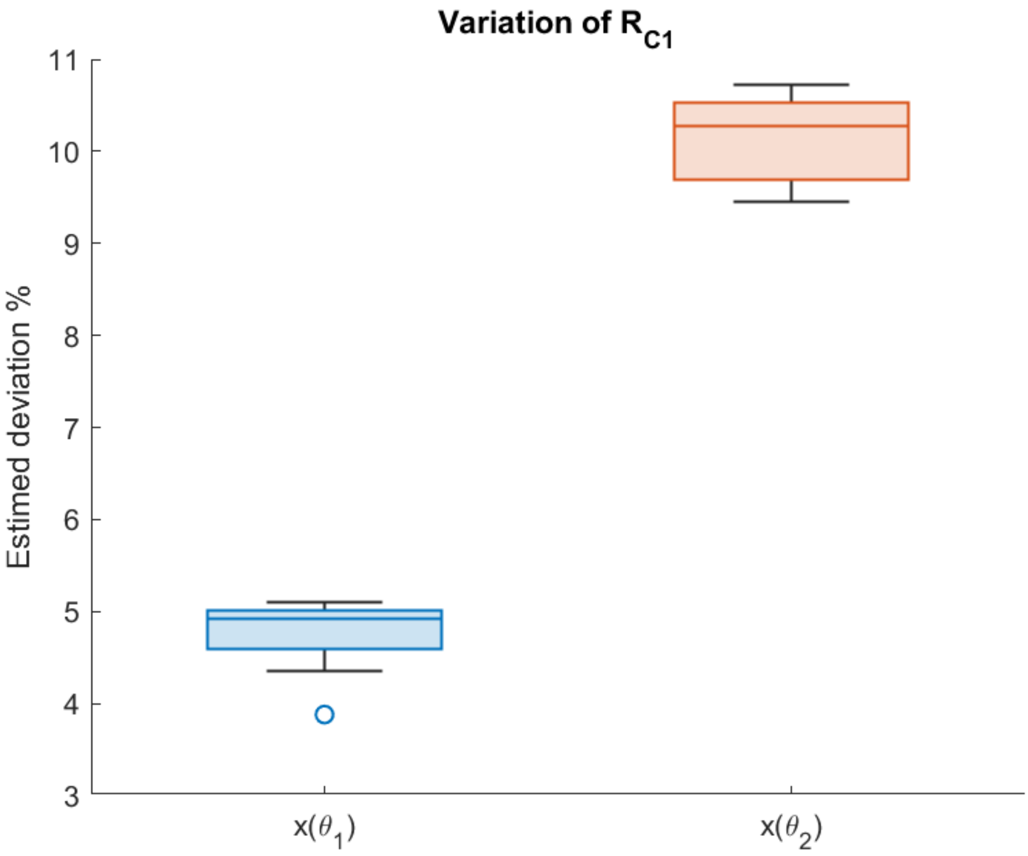


Figure 43 Estimation result for R_{c1}

From the graph presented, it is important to notice that the algorithm is able to identify this parameter with an extremely good tolerance. Even considering the outlier, none of the results crossed the 1% deviation from the solution of the problem. This is an exceptionally positive result, considering how high the threshold for failure is for this component.

Focusing now on how the other failure indicator behaved. The estimation of the ESR of the second capacitance R_{c2} is depicted in *Figure 46*.

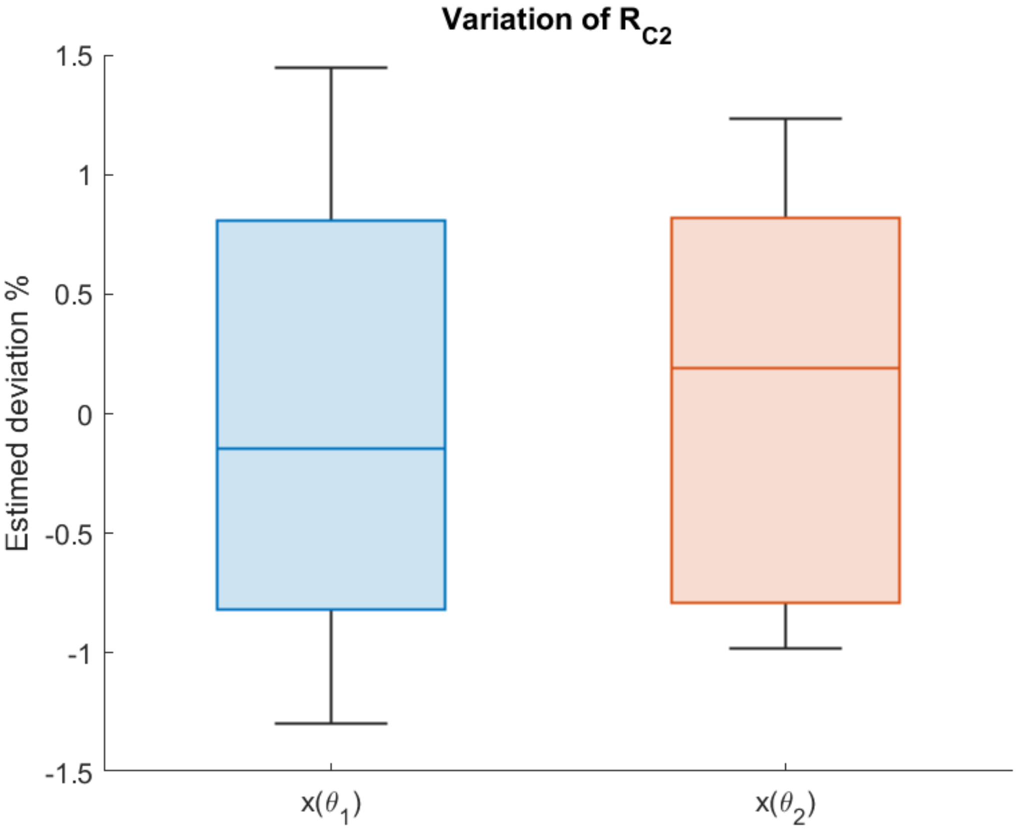


Figure 44 Estimation result for R_{c2}

It is possible to notice that the results obtained for R_{c2} did not present any noticeable deviation from the solution of the problem, suggesting that the algorithm is able to pinpoint the component that is the root cause of the increased cost function and provide results without meaningful correlation with the other parameter. This is a very promising result since it shows that it is possible to identify those two parameters independently. In the identification of this ESR parameter, none of the results crossed the 1% deviation from the solution of the problem.

The data of the identification of the R_{on} are presented in the graph in *Figure 47*. R_{on} from the test performed, proved to be the most challenging to identify. The identification of R_{on} is limited compared to other parameters because its variation did not affect the output characteristics as deeply as the changes in the ESR.

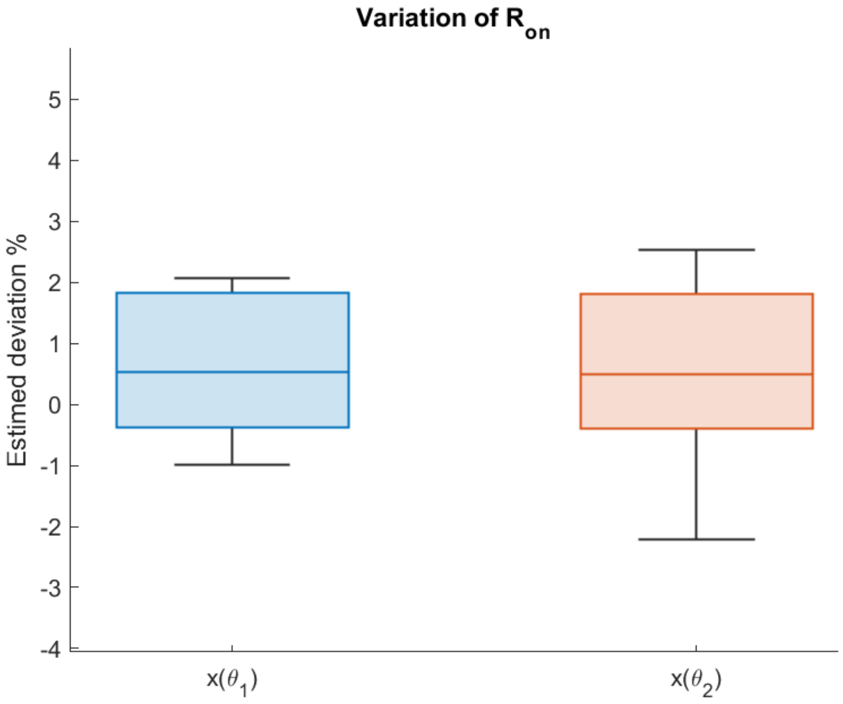


Figure 45 Estimation result for R_{on}

The results of the identification for R_{on} are still very positive. The mean value of between the results correctly indicates the steadiness of the parameter throughout the test. On the other hand, the distribution of the result has noticeably a larger variance when compared with the estimation of the capacitor ESR.

The overall results of this test are positive, especially when compared to the order of magnitude of the deviation that is required to estimate. As previously stated, the expected deviation for the ESR of a capacitor before a fault can reach a deviation as large as 278%. From the accuracy displayed in this test, it is possible to state that the used method is more than capable of identifying the aging of the capacitor. It has to be noted that the results presented were possible because even a small deviation in the ESR was able *to harshly* impact the output characteristics. This is a crucial factor that definitely helped the identification process by making the cost function steeper around the solution which also improves the ability to discern in the real case the inevitable noise in the measurement from a change in said parameter.

4.6.2 MOSFET identification

Unlike the previous test with the gradual increase of the ESR evaluating the quality of the estimation for R_{on} were not as successful. In the initial test, the deviation of R_{on} was, as in the previous test, of five and ten percent. This initial test, were a failure since the algorithm was unable to reliably detect the deviation in the parameter.

Since it was not possible to detect finer parameter deviation, the algorithm was tested to prove if the system was at least accurate enough to detect the failure of the component. In [33], a 50% increase in R_{on} is defined as a warning threshold. With this objective of proving if the method was able to detect if R_{on} cross the warning threshold, it was decided to test the algorithm by identifying the following four ripple characteristics that were simulated: The first ripple that was simulated was the baseline where all the parameters were the nominal ones, the second presented an increased R_{on} equal to half the warning threshold, the third test presented an increased R_{on} equal to the warning threshold and lastly the fourth test presented an increased R_{on} that exceeds the warning threshold by 50%. The arrays of the parameters used in the test are presented in *Table 10.*

	θ_0	θ_1	θ_2	θ_3
L	33μH	33μH	33μH	33μH
C_1	20μF	20μF	20μF	20μF
C_2	440μF	440μF	440μF	440μF
R_L	60mΩ	60mΩ	60mΩ	60mΩ
R_{C1}	60mΩ	60mΩ	60mΩ	60mΩ
R_{C2}	65mΩ	65mΩ	65mΩ	65mΩ
R_{on}	0.100Ω	0.125Ω	0.150Ω	0.175Ω

Table 10 vector of parameters used in the test

As in the previous tests the data are presented in a graph where the point on the x-axis is meant to represent a point in time where the parameters are set to $x(\theta_n)$, and in the y-axis are plotted the results of the identification in the form of the normalized deviation of the parameters to the respective nominal values.

Presenting now the result for the identification of R_{on} on *Figure 48*, from this first graph it is possible to observe that even if the overall variance of the result is much higher than the one found in the previous test, the average value of the different result is overall consistent with the parameters of the solution.

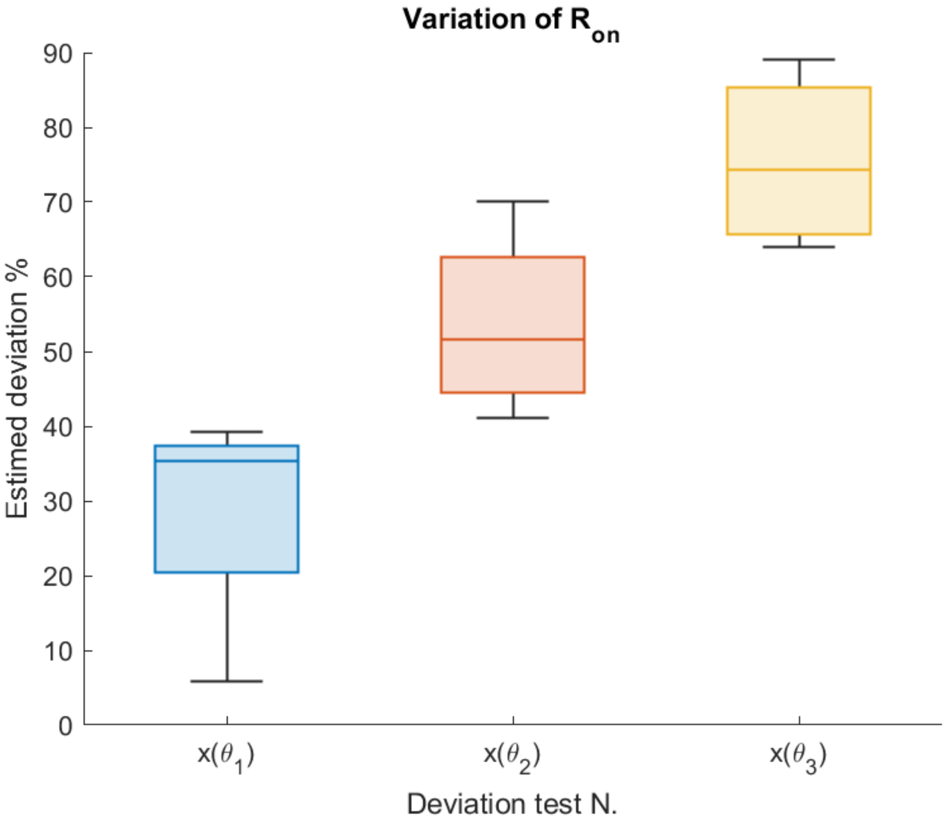


Figure 46 Estimation result for R_{on}

Observing the data is possible to notice that, with the exception of some outliers, the results are spread mostly within a 15% deviation. Even if the tolerances for the identification are so high, the mean of the result consistently follows the solution of the identification.

Plotting together the mediated results with the actual solution for the identification, as was made in *Figure 49*, it is possible to see that the method is a viable way to mitigate the inaccuracy of the identification. Most importantly this system in have the capabilities to detect if the drift of R_{on} exceed the warning threshold that was proposed in [33].

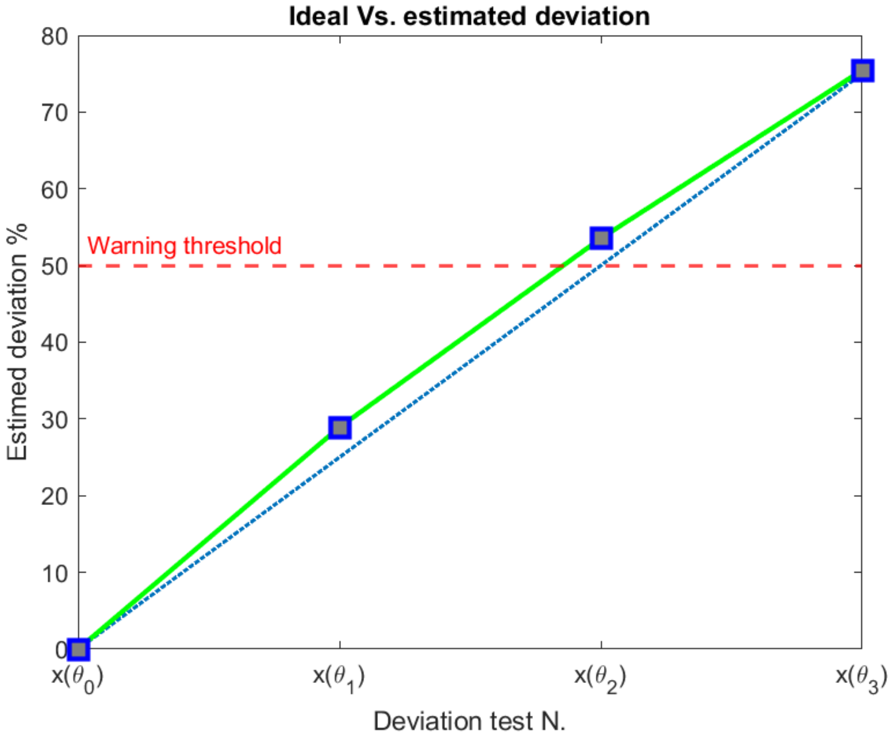


Figure 47 Mean values for the identifications of R_{on} *(blue dot) compared with the ideal result (dotted blue line)*

We analyze now the other parameter starting from R_{c1}. As can be noticed in the graph of *Figure 50*, also in this test, the result of the identification of the ESR parameter was extremely good. Not only did the mean value of the various identifications differ by marginal value from the solution, but also the variance of the result was extremely low.

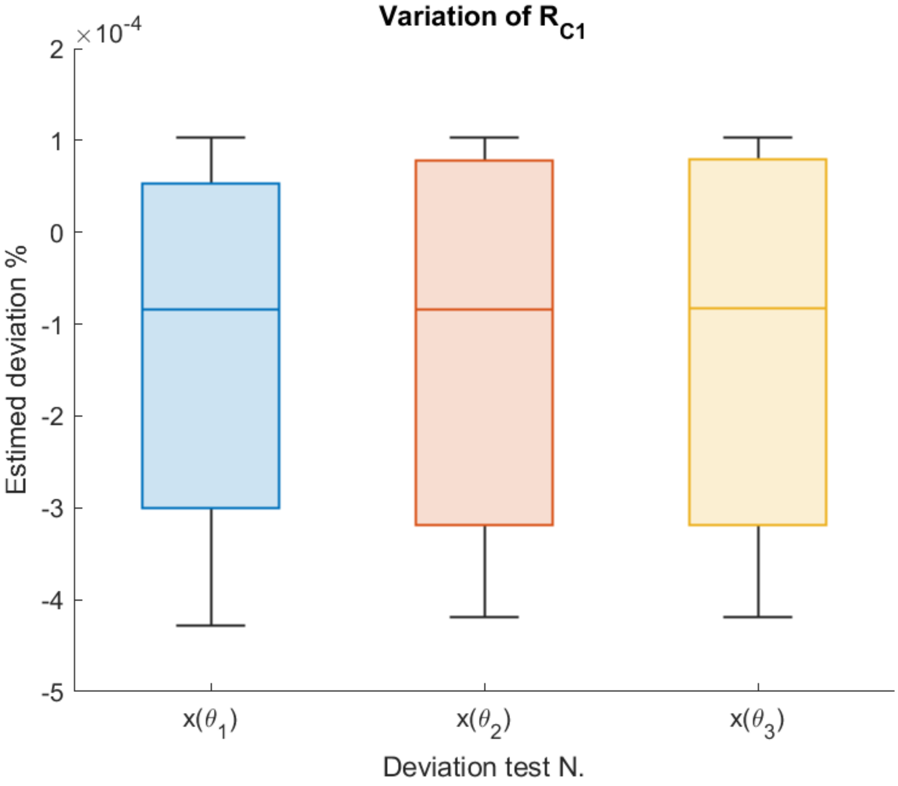


Figure 48 Estimation result for R_{c1}

Finally, we study the last failure indicator remaining, R_{c2}. It is possible to notice that similarly to the result found for R_{c1} the mean value of the results differs from the solution by a neglectable margin. Also, the variance that characterizes the distribution of the solution is extremely low with values that are consistent with the one found for R_{c1}.

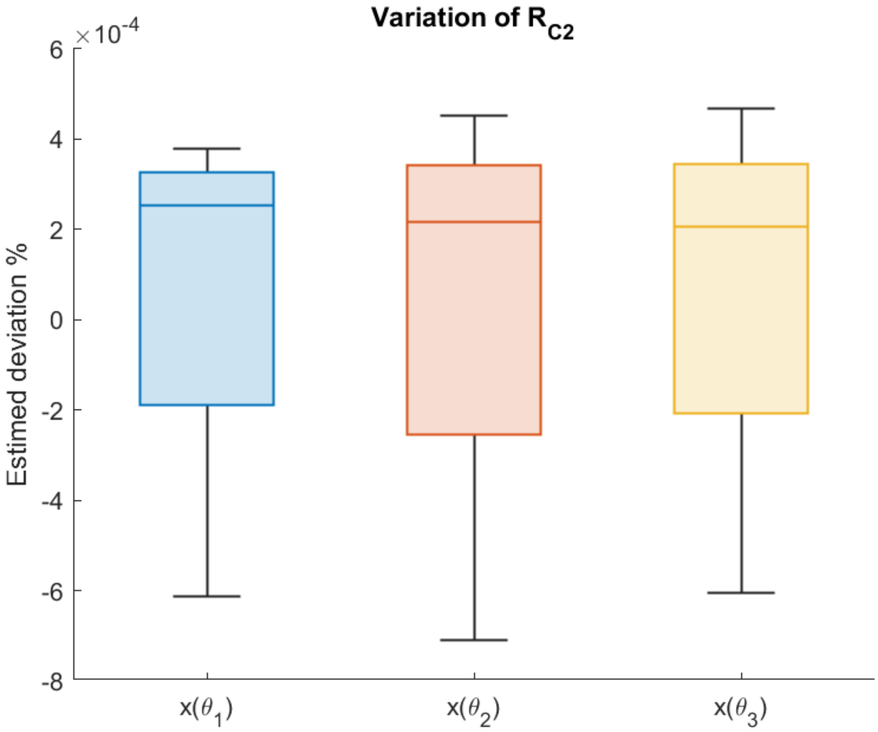


Figure 49 Estimation result for R_{c2}

Chapter 5

Application on a single-phase inverter

Since the overall results for the buck converter were deemed positive, a further investigation of the algorithm performance with a more complex system was made. For this test, the power converter chosen for the identification process was a single-phase inverter. This test was performed in order to push the capabilities of the algorithm and confirm if it was able to extract meaningful data for the fault parameters in a system with higher complexity.

As in the previous test, a Simulink model was implemented to generate the output characteristic. The data were then aligned, and a cost function was built on the differences between "target" and simulated characteristics. The data alignment applied for this test was the one developed for the previous buck converter test. For the preliminary test only the basic $f_{RSS}\ x$ cost function described in the *equation 2.1* was used. The inverter model used for this test is depicted in *Figure 52.*

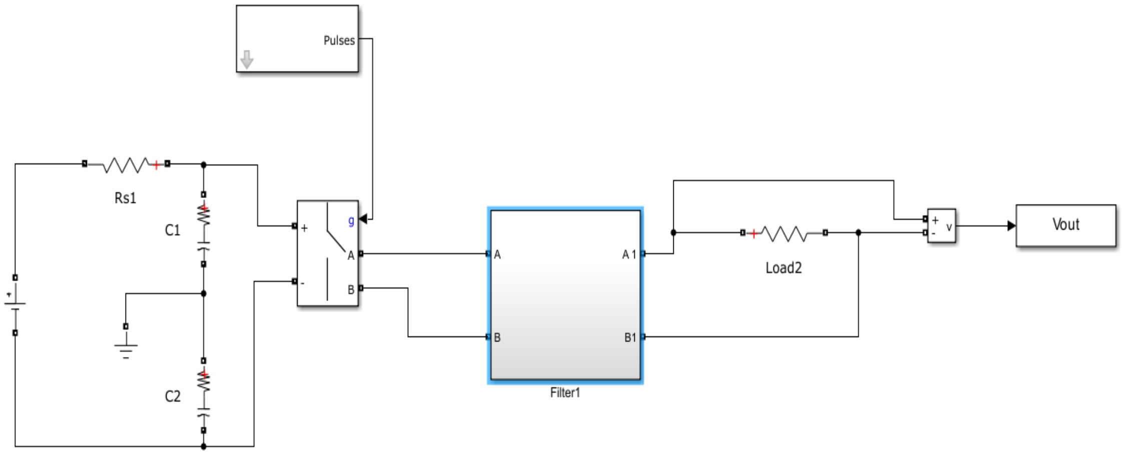


Figure 50 Single-phase inverter Simulink model

The filter block present in the F*igure 52* was modelled as follows:

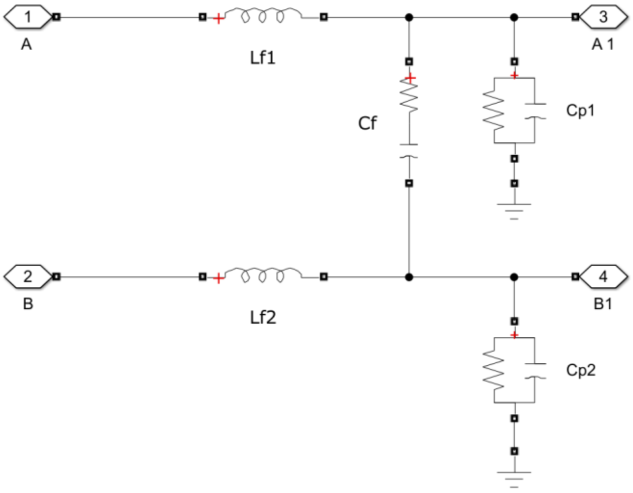


Figure 51 Filter schematic in Simulink

The output characteristic of this model is presented in the graph of *Figure 54*.

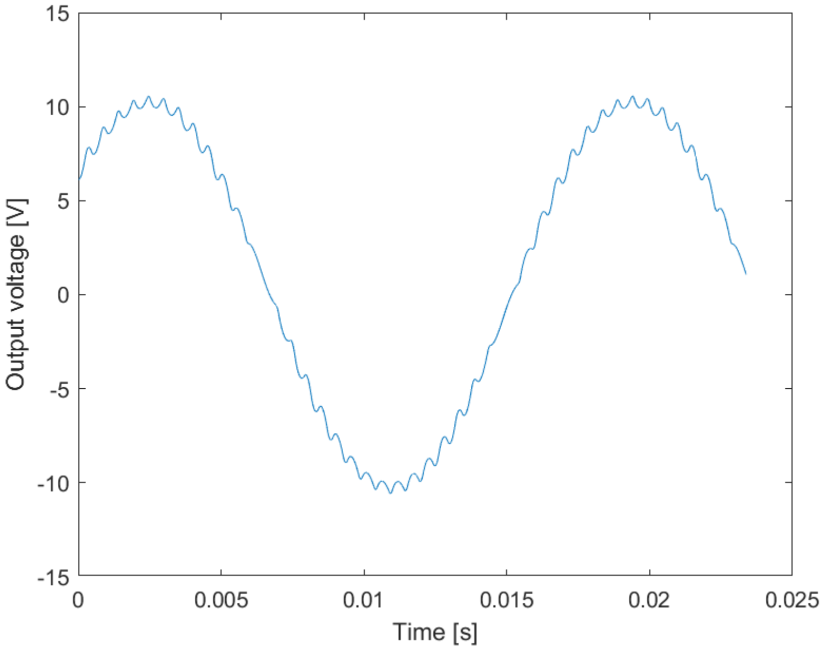


Figure 52 Output characteristic of the single-phase inverter model

It is possible to notice from *Figure 54*, that this type of device presents a ripple that the filter block was unable to completely smooth out. To verify the performance of this setup, as in the previous test, the first set of identification problems generated was aimed to emulate a drift in the ESR parameter on one of the filter capacitors. The ESR chosen to be identified was the one relative to the capacitor C_f. This choice was made since, among the capacitors present in the filter block, C_f is the one with the largest ESR. Any shift in this parameter will therefore have a higher impact on the output ripple. Therefore, should make this fault parameter the easiest to identify. The vectors of parameters for the identification problem are depicted in *Table 11*.

	θ_0	θ_1	θ_2	θ_3
L_{f1}	2mH	2mH	2mH	2mH
L_{f2}	2mH	2mH	2mH	2mH
C_f	35μF	35μF	35μF	35μF
R_{Cf}	60mΩ	60mΩ + 15mΩ	60mΩ + 30mΩ	60mΩ + 60mΩ
C_{p1}	300pF	300pF	300pF	300pF
C_{p2}	300pF	300pF	300pF	300pF
R_{cp1}	3000Ω	3000Ω	3000Ω	3000Ω
R_{cp2}	3000Ω	3000Ω	3000Ω	3000Ω

Table 11 Set of parameters for the different identifications

As done for the previous identification test, each problem was iterated ten times. The initial conditions for the first identification were generated by perturbating the nominal values of the system with Gaussian noise. The variance of said Gaussian distribution was $\sigma^2 = 2.7\overline{7} \times 10^{-4}$ and mean $\mu = 0$. The variance was chosen in order to have $3\sigma = 0.05$ meaning that 99.7% of the initial conditions generated did not exceed a 5% deviation from the result from which they were derived. For all the successive identifications, the mediated result of the previous identification is used as a base for the initial conditions. The results found are depicted in the graph of *Figure 55.*

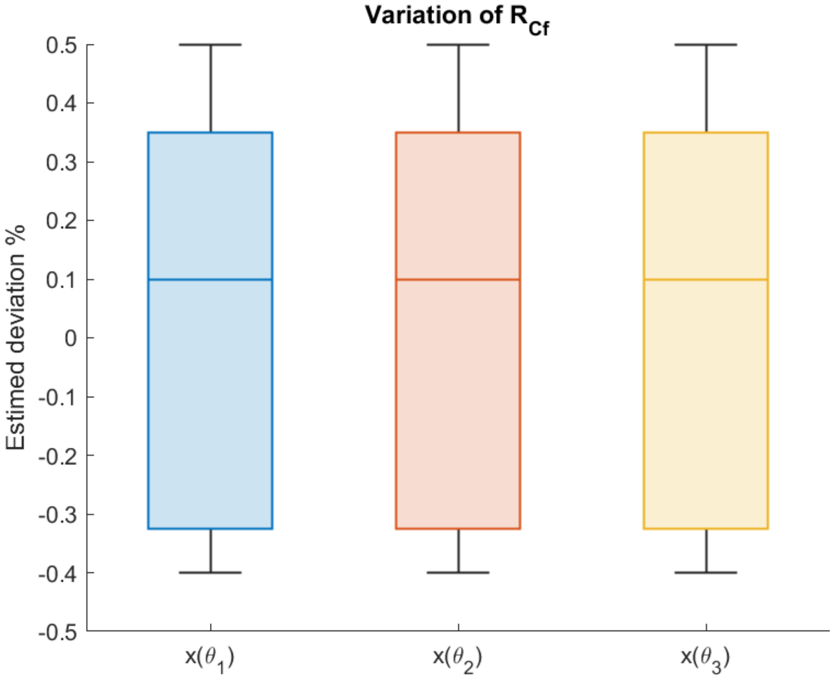


Figure 53 Identification result of R_{c2}

Unfortunately, as highlighted in *Figure 55*, the algorithm was not able to identify any shift in the parameter values between the different tests. Observing the variance of the results, one might conclude the presence of a local minima to which the parameter converges. Studying the result, it was possible to conclude that this was not the case. The variance in the result was so limited only because the output result did not present any meaningful shift from the values from which the optimization started. Therefore, the distribution of the result just mirrors the distribution of the initial conditions.

Chapter 6
Conclusions

In conclusion, this thesis presented an effective identification method aimed at assessing the health of the most vulnerable component present in a power electronic system, with the objective of providing a valuable tool to improve the reliability of the predictive maintenance performed on this type of system.

The algorithm implemented measures the input and output characteristics of the system to be identified, it then implements a model to provide a simulated output characteristic for the measured input. A cost function is then built to highlight the difference between the measured and simulated output characteristics. Finally, an optimizer was implemented to minimize the cost function by tuning the values of the component of the model.

A simple RLC circuit was initially used to perform a thorough analysis on how to build a cost function that improves the conversion capabilities of the optimization algorithm. Once the cost function was tuned, different optimization algorithms were benchmarked. The accuracy of the results and computational efficiency in solving the optimization problems were the key aspects analyzed to compare the performance of the different algorithms. To provide a reference baseline performance, the non-linear least square algorithm was chosen. This choice was made since it is the most used algorithm in literature. The optimizers tested were patternsearch, PTO and Bayes opt. The result of this benchmark highlighted that implementing a cost function that leverages some distinct feature of the output characteristic will result in an improvement in the convergence capabilities. Among the algorithms tested the patternsearch algorithm proved to be the most capable one, it was distinguished from the others not only for its efficiency and accuracy but also for its ability to solve a broader set of problems. The identification algorithm was then tested for a more complex system, a buck converter. In those tests, the capabilities of the algorithm to identify the drift in values of the failure parameter of the capacitor and MOSFET were tested. The algorithm proved its effectiveness in identifying the ESR values with minimal tolerances of around 1%, this level of accuracy is much higher than the one requested to assess the health of the component. The identification result for the R_{on} of the MOSFET presented tolerances around the 15% mark, this level of accuracy is not enough to make this algorithm a reliable and robust solution for the health assessment of the system. To mitigate the inaccuracy of the identification multiple identifications were performed with different initial conditions and the output result of the identification was the mean of those identifications. From the test performed implementing the mediated result allowed to correctly identify when the safety threshold for the Ron value is crossed.

The implementation of a Simulink model to generate the output characteristics severely impaired the efficiency of the algorithm. One identification required 8 hours to perform, which limited the number of tests that could be performed. The results obtained are therefore a proof of concept rather than a real benchmark on the performance of this method.

Future investigations should be performed with more efficient software in order to have a more reasonable computational burden and be able to provide a more extensive set of results. A larger dataset would allow for more meaningful data to study in depth the benefit of a modified cost function and explore different tuning methods in more complex systems. Finalizing an optimized version of the software would allow to provide the requirement for the on-board hardware that will perform the computations.

Bibliography

[1] Gabriel Rojas-Dueñas, Jordi-Roger Riba, Member, IEEE, and Manuel Moreno-Eguilaz,” A Digital Twin Based Estimation Method for Health Indicators of DC–DC Converters” IEEE Trans. Power Electron, VOL. 36, no. 1, pp. 654-661, January 2021.

[2] Iannuzzo, Francesco & Abbate, Carmine & Busatto, Giovanni. (2014). Instabilities in Silicon Power Devices: A Review of Failure Mechanisms in Modern Power Devices. Industrial Electronics Magazine, IEEE. 8. 28-39

[3] Saeed Rahimpour, Hadi Tarzamni, Naser Vosoughi Kurdkandi, Oleksandr Husev, Dmitri Vinnikov, and Farzad Tahami,” An Overview of Lifetime Management of Power Electronic Converters” 10.1109/ACCESS.2022.3214320, October 2022.

[4] A. Malik, A. Haque and K. V. Satya Bharath, "Fault Tolerant Inverter for Grid Connected Photovoltaic System," 2022 IEEE International Conference on Power Electronics, Smart Grid, and Renewable Energy (PESGRE), Trivandrum, India, 2022, pp. 1-6

[5] H. Wang, M. Liserre and F. Blaabjerg, "Toward Reliable Power Electronics: Challenges, Design Tools, and Opportunities," in IEEE Industrial Electronics Magazine, vol. 7, no. 2, pp. 17-26, June 2013

[6] Liang, J., Zhang, K., Al-Durra, A., Muyeen, S., & Zhou, D, “A state-of-the-art review on wind power converter fault diagnosis”, Energy Reports, 8, pp 5341-5369, 2022.

[7] Shi Zheng-Yu, Lu Yu-Dong, Ning Tao, Li Meng-Qi, Feng Jing-Dong and Zhou Zhen-Wei, "The real-time fault diagnosis of electrolytic filter capacitors in switching mode power supply," Proceedings of the 20th IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA), Suzhou, China, 2013, pp. 662-665

[8] S. Chen, S. Wang, P. Wen and S. Zhao, "Digital Twin for Degradation Parameters Identification of DC-DC Converters Based on Bayesian Optimization," 2021 IEEE International Conference on Prognostics and Health Management (ICPHM), Detroit (Romulus), MI, USA, 2021, pp. 1-9

[9] Yingzhou Peng, IEEE, Shuai Zhao, Member IEEE, and Huai Wang,” Nonlinear Least Squares Optimization for Parametric Identification of DC–DC Converters” IEEE Trans. Power Electron, VOL. 36, no. 2, pp. 2105-2117 February 2021.

[10] Liu, He & Xia, Min & Williams, Darren & Sun, Jianzhong & Hongsheng, Yan. (2022). Digital Twin-Driven Machine Condition Monitoring: A Literature Review. Journal of Sensors. 2022.

[11] Zhong, D., Xia, Z., Zhu, Y., & Duan, J. Overview of predictive maintenance based on digital twin technology. Heliyon, 9(4), e14534, pp 1 -23 2023.

[12] Negri, E., Fumagalli, L., & Macchi, M. (2016). A Review of the Roles of Digital Twin in CPS-based Production Systems. Procedia Manufacturing, 11, pp. 939-948.

[13] Yingzhou Peng, Huai Wang," Application of Digital Twin Concept in Condition Monitoring for DC-DC Converters", IEEE Energy Conversion Congress and Exposition (ECCE), Baltimore, MD, USA, 2019, pp. 2199-2204

[14] L. Ferreira Costa and M. Liserre, "Failure Analysis of the dc-dc Converter: A Comprehensive Survey of Faults and Solutions for Improving Reliability," in IEEE Power Electronics Magazine, vol. 5, no. 4, pp. 42-51, Dec. 2018.

[15] S. Yang, A. Bryant, P. Mawby, D. Xiang, L. Ran and P. Tavner, "An industry-based survey of reliability in power electronic converters," 2009 IEEE Energy Conversion Congress and Exposition, San Jose, CA, USA, 2009, pp. 3151-3157

[16] R. Orsagh, D. Brown, M. Roemer, T. Dabnev and A. Hess, "Prognostic health management for avionics system power supplies," 2005 IEEE Aerospace Conference, Big Sky, MT, USA, 2005, pp. 3585-3591.

[17] Wuchen Wu, M. Held, P. Jacob, P. Scacco and A. Birolini, "Thermal stress related packaging failure in power IGBT modules," Proceedings of International Symposium on Power Semiconductor Devices and IC's: ISPSD '95, Yokohama, Japan, 1995, pp. 330-334.

[18] Z. Ni, X. Lyu, O. P. Yadav, B. N. Singh, S. Zheng and D. Cao, "Overview of Real-Time Lifetime Prediction and Extension for SiC Power Converters," in IEEE Transactions on Power Electronics, vol. 35, no. 8, pp. 7765-7794, Aug. 2020

[19] Robert Borgovini, Stephen Pemberton, Michael Rossi," Failure Mode, Effects and Criticality Analysis (FMECA)", April 1993, pp. 1-127

[20] Khan, Shahamat & Wen, Huiqing. (2021). A Comprehensive Review of Fault Diagnosis and Tolerant Control in DC-DC Converters for DC Microgrids. IEEE Access. PP. 1-28.

[21] Riba, J., Bogarra, S., & Garcia, A. (2018). Parameter Identification of DC-DC Converters under Steady-State and Transient Conditions Based on White-Box Models. Electronics, 7(12), 393.

[22] B. H. Lin, J. T. Tsai and K. L. Lian, "A Non-Invasive Method for Estimating Circuit and Control Parameters of Voltage Source Converters," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 66, no. 12, pp. 4911-4921, Dec. 2019

[23] J. W. Kimball and P. T. Krein, "A current-sensorless digital controller for active power factor correction control based on Kalman filters," 2008 Twenty-Third Annual IEEE Applied Power Electronics Conference and Exposition, Austin, TX, USA, 2008, pp. 1328-1333

[24] M. Ahmeid, M. Armstrong, S. Gadoue, M. Al-Greer and P. Missailidis, "Real-Time Parameter Estimation of DC–DC Converters Using a Self-Tuned Kalman Filter," in IEEE Transactions on Power Electronics, vol. 32, no. 7, pp. 5666-5674, July 2017

[25] M. Algreer, M. Armstrong and D. Giaouris, "Active Online System Identification of Switch Mode DC–DC Power Converter Based on Efficient Recursive DCD-IIR Adaptive Filter," in IEEE Transactions on Power Electronics, vol. 27, no. 11, pp. 4425-4435, Nov. 2012

[26] M. Al-Greer, M. Armstrong, M. Ahmeid and D. Giaouris, "Advances on System Identification Techniques for DC–DC Switch Mode Power Converter Applications," in IEEE Transactions on Power Electronics, vol. 34, no. 7, pp. 6973-6990, July 2019,

[27] Abramson, Mark. (2003). Pattern Search Algorithms for Mixed Variable General Constrained Optimization Problems. PhD thesis

[28] Audet, Charles & Dennis, J. (2006). Mesh Adaptive Direct Search Algorithms for Constrained Optimization. SIAM Journal on Optimization. 17. 188-217.

[29] Beiranvand, V., Hare, W., & Lucet, Y. (2017). Best practices for comparing optimization algorithms. Springer http://dx.doi.org/10.1007/s11081- 017-9366-1

[30] Y. Yan, Q. Liu and Y. Li, "Paradox-Free Analysis for Comparing the Performance of Optimization Algorithms," in IEEE Transactions on Evolutionary Computation, vol. 27, no. 5, pp. 1275-1287, Oct. 2023

[31] Moré, Jorge & Wild, Stefan. (2009). Benchmarking Derivative-Free Optimization Algorithms. SIAM Journal on Optimization. 20.

[32] M. Heydarzadeh, S. Dusmez, M. Nourani and B. Akin, "Bayesian remaining useful lifetime prediction of thermally aged power MOSFETs," 2017 IEEE Applied Power Electronics Conference and Exposition (APEC), Tampa, FL, USA, 2017, pp. 2718-2722

[33] Du, Xinghao & Meng, Jinhao & Liu, Kailong & Zhang, Yingmin & Wang, Shunli & Peng, Jichang & Liu, Tianqi. (2021). Online Identification of Lithium-ion

Battery Model Parameters with Initial Value Uncertainty and Measurement Noise.

[34] Ammar Mohammed, Rania Kora," A comprehensive review on ensemble deep learning: Opportunities and challenges", Journal of King Saud University - Computer and Information Sciences, Volume 35, Issue 2,2023, Pages 757-774.

[35] Texas instruments," Using the TPS40200", TPS40200 datasheet, April 2009, Rev.1.

[36] Onsemi," MOSFET-P-Channel POWERTRENCH, Logic Level", FDC5614P datasheet, July 2022, Rev.4.

[37] Onsemi," Surface Mount Schottky Power Rectifier", MBRS360T3G datasheet, June 2024, Rev.15.

[38] J. Poon, P. Jain, C. Spanos, S. K. Panda and S. R. Sanders, "Fault Prognosis for Power Electronics Systems Using Adaptive Parameter Identification," in IEEE Transactions on Industry Applications, vol. 53, no. 3, pp. 2862-2870, May-June 2017

[39] Murali Yeddanapud. (2017). master class speeding up Simulink applications. Mathworks

[40] Box chart (box plot), MathWorks, https://it.mathworks.com/help/matlab /ref/boxchart.html#mw_e9da6873-07c2- 42c3-9cce-266f01fd1dd3.