



PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

IMPLEMENTING A POWER HARDWARE IN-THE-LOOP PLATFORM USING THE DAMPING IMPEDANCE METHOD

FELIPE ANTONIO CHAPARRO PÉREZ

Tesis para optar al grado de
Magíster en Ciencias de la Ingeniería

Profesor Supervisor:
JAVIER PEREDA TORRES

Santiago de Chile, Octubre 2020

© MMXX, FELIPE ANTONIO CHAPARRO PÉREZ



PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

IMPLEMENTING A POWER HARDWARE IN-THE-LOOP PLATFORM USING THE DAMPING IMPEDANCE METHOD

FELIPE ANTONIO CHAPARRO PÉREZ

Miembros del Comité:

JAVIER PEREDA TORRES

Javier Pereda
Javier Pereda (Oct 27, 2020 14:50 ADT)

FELIPE NUÑEZ

Felipe Nuñez
Felipe Nuñez (Oct 27, 2020 23:09 ADT)

TOMÁS ÁVILA

Tomás Ávila

DIEGO LOPEZ-GARCIA

Diego López-García
Diego López-García (Oct 28, 2020 09:35 ADT)

Para completar las exigencias del grado de
Magíster en Ciencias de la Ingeniería

Santiago de Chile, Octubre 2020

© MMXX, FELIPE ANTONIO CHAPARRO PÉREZ

*A mi familia, en particular a mi madre
Soledad y a Ester, quienes me han
cuidado desde siempre.*

AGRADECIMIENTOS

En primer lugar me gustaría dar las gracias a mi profesor guía, Javier Pereda, por su paciencia, apoyo, consideración. Por aceptarme como su alumno y guiarme, incluso cuando los temas de investigación resultaban eran nuevos para ambos.

Quisiera agradecer también a el apoyo de los proyectos de Fondap y fondecyt mediante los proyectos: ANID/FONDAP/15110019, ANID/PIA/ACT192013 y ANID/ FONDECYT/ 1171142. Agradezco a mis amigos, por su apoyo y alegría. Por estar ahí para mí tanto en los momentos felices como en los más complicados. Y sobre todo, por quererme y aceptarme como soy.

Doy las gracias a Engie Laborelec, tanto a su equipo chileno como belga. Agradezco profundamente a Sebastián Michels por creer en mí, darme la oportunidad de realizar la práctica profesional con la compañía y por convencerlos de patrocinar mi tesis. A Tomás Ávila, por ser un gran apoyo, por sus consejos y ayuda, además de por aceptar ser parte de la comisión evaluadora. También agradezco a Luis Valenzuela, a Antonio Alarcón y a todo el personal de Laborelec Chile.

Agradezco sinceramente al equipo de Power Systems en Laborelec Bélgica por hacer mi estadía en Bruselas un tiempo muy ameno y enriquecedor. Agradezco muy especialmente a Stijn Uytterhoeven, Olivier Ducarme, Loic Maudoux y Christ Scheldeman por su apoyo y ayuda. Por permitirme realizar los experimentos y siempre estar dispuestos a ayudar cuando lo requerí. Una mención especial merece Laura Terán, cuyo apoyo fue crucial en la última etapa de este trabajo, durante la pandemia.

Finalmente, quisiera agradecer a mi familia, a mi madre Soledad, que siempre me ha apoyado, querido, protegido y entendido; y a Ester, quién ha estado ahí para mí desde que tengo memoria. Entre ambas, no podría pedir una mejor familia.

GENERAL INDEX

AGRADECIMIENTOS	IV
FIGURE INDEX	VIII
TABLE INDEX	XV
RESUMEN	XVI
ABSTRACT	XVII
1. INTRODUCTION	1
1.1. The role of power hardware in-the-loop in testing	2
1.2. State-of-art of PHIL	7
1.3. Original contribution of the thesis	15
1.3.1. Objectives	16
1.3.2. Hypothesis	17
1.3.3. Methodology	17
1.3.4. Organisation of the thesis	19
2. Interface algorithms: An implementation oriented review	20
2.1. Introduction	20
2.2. Ideal transformer method (ITM)	21
2.2.1. Stability considerations	22
2.2.2. Accuracy considerations	28
2.3. Ideal transformer method with low pass filter	32
2.3.1. Stability considerations	32
2.3.2. Accuracy considerations	34
2.4. Partial circuit duplication (PCD)	37
2.4.1. Stability considerations	38

2.4.2.	Accuracy considerations	43
2.5.	Damping Impedance method (DIM)	48
2.5.1.	Stability considerations	49
2.5.2.	Accuracy considerations	57
2.6.	Effects of an active HUT	64
2.6.1.	Effects on an active HUT on stability	66
2.6.2.	Effects on an active HUT on accuracy	67
2.7.	Chapter conclusion	70
3.	Impedance identification	72
3.1.	Introduction	72
3.2.	Identification procedure overview	73
3.3.	Data gathering	74
3.4.	Correlation functions and power spectral density	75
3.5.	Spectral analysis	78
3.6.	Windowing	81
3.7.	Excitation	83
3.7.1.	Sine and sum of sines	83
3.7.2.	Step function	84
3.7.3.	White noise & Pseudo Random Binary Sequences (PRBS)	85
3.8.	Spectral analysis: implementation and offline results	88
3.9.	Parametric identification	93
3.10.	Levenberg-Marquardt method	97
3.11.	Chapter conclusion	103
4.	Software implementation & simulation results	104
4.1.	Introduction	104
4.2.	Opal-RT model structure	105
4.3.	Software implementation: Electrical simulation	108

4.4.	Software implementation: Identification Routine	113
4.5.	Software implementation: Measurements	114
4.6.	Simulation results	116
4.6.1.	Passive load	116
4.6.2.	Active load	127
4.7.	Problems of spectral analysis in the active load case	137
4.8.	Chapter conclusion	143
5.	Hardware implementation & experimental results	145
5.1.	Introduction	145
5.2.	Guidelines for programming for real time with Matlab Simulink and RT-LAB 145	
5.2.1.	Discrete solvers and algebraic loops	145
5.2.2.	Using code compatible with code generation	147
5.3.	Changes to subsystems	148
5.4.	Physical components of PHIL platform	151
5.4.1.	OPAL-RT target: OP5600	151
5.4.2.	Voltage amplifier	152
5.4.3.	Delta electronica DC source	154
5.4.4.	HUTs	155
5.5.	Testing: Passive load	156
5.6.	Testing: Solar inverter	164
5.7.	Chapter conclusion	176
6.	Conclusions	177
	BIBLIOGRAFY	180
	ANEXO A. Additional Resources	190

FIGURE INDEX

1.1. Testing methodologies	2
1.2. LVRT voltage profile. [2020] IEEE	3
1.3. CHIL and PHIL architecture	5
1.4. Amplifier size comparison	10
1.5. ITM interface algorithm	11
1.6. DIM algorithm	12
1.7. Block diagram of the WSI technique. [2017] IEEE	14
1.8. Block diagram of the postprocessing routine to identify the parametric impedance. Implemented in LabVIEW [2017] IEEE	15
2.1. ITM interface algorithm	22
2.2. ITM algorithm block diagram	23
2.3. Comparison between stable and unstable cases of ITM algorithm	25
2.4. Comparison between stable and unstable cases of ITM algorithm. Nyquist diagram	26
2.5. Difference between real system and system with delay approximation.	28
2.6. Naturally coupled system	29
2.7. NCS vs ITM	30
2.8. ITM. Deviations with respect to NCS	31
2.9. ITM method with lowpass filter. Block diagram	32
2.10. Comparison between ITM and ITM with lowpass filter IAs	33
2.11. Comparison between ITM and ITM with lowpass filter IAs.	34

2.12. Closed loop response of system with low pass filter compared to ITM and NCS systems	35
2.13. ITM with low pass filter. Deviations with respect to NCS	36
2.14. PDC algorithm	37
2.15. PCD block diagram: forward path	38
2.16. PCD block diagram: complete	38
2.17. PCD. Comparison between purely resistive and purely inductive Z_{ab}	40
2.18. Nyquist contours of PCD algorithm. Multiple values for Z_{ab}	42
2.19. PCD vs ITM vs ITM with lowpass filter	42
2.20. Simplified control loop	43
2.21. PCD accuracy. Multiple values	45
2.22. PCD. Multiple values of Z_{ab} . Deviations with respect to NCS	46
2.23. PCD accuracy vs other interface algorithms	47
2.24. PCD. Deviations with respect to NCS	47
2.25. DIM algorithm	49
2.26. DIM forward path	50
2.27. DIM left side solved by superposition	50
2.28. DIM control block diagram	51
2.29. DIM control block diagram: modified	52
2.30. Nyquist contours of DIM algorithm. Multiple values for Z_{DIM}	55
2.31. Comparison between the different interface algorithms	56
2.32. Comparison between the NCS and DIM	59
2.33. DIM. Multiple value of Z_{DIM}	61
2.34. DIM. Multiple value of Z_{DIM} . Deviations with respect to NCS	61
2.35. Comparison between the different values of Z_{ab}	62

2.36. Comparison between different IAs	62
2.37. Comparison between different IAs. Deviation with respect to the NCS	63
2.38. Active HUT	64
2.39. Block diagram of ITM IA with an active HUT	65
2.40. Block diagram of PCD IA with an active HUT	65
2.41. Block diagram of DIM IA with an active HUT	65
2.42. Naturally coupled system with an active HUT	67
3.1. Data gathering environment	75
3.2. Spectrum of several input signals	85
3.3. LFSR implementing PRBS7: $x^7 + x^6 + 1$	88
3.4. Identification circuit for case 1: RL	90
3.5. Identification circuit for case 2: RLC	91
3.6. Results for RL filter identification	91
3.7. Results for RLC filter identification	92
3.8. Results for RL filter identification	96
3.9. Results for RLC filter identification	96
3.10. Least square ID. N = 2000	97
3.11. RL circuit ID with LM method. N = 2000	102
3.12. RL circuit ID with LM method. N = 10000	102
4.1. PHIL platform structure	105
4.2. OPAL RT model structure	106
4.3. Software structure	107
4.4. Top view of Simulink model	108
4.5. Overview of subsystem SM computations	110

4.6. Implementation of the DIM algorithm	111
4.7. Implementation of a variable impedance	111
4.8. Amplifier and HUT	112
4.9. Implementation of data sampler	113
4.10. Implementation of the ID routine	114
4.11. Implementation of the DIM algorithm	115
4.12. Simplified diagram of the simulation	117
4.13. Identification process. First 2 attempts	118
4.14. Identification process. First 10 attempts	119
4.15. Simulation voltage. RL load	120
4.16. Simulation current. RL load	121
4.17. Active power. RL load	121
4.18. Reactive power. RL load	122
4.19. Voltage relative error. RL load	123
4.20. Voltage relative error zoom. RL load	123
4.21. Current relative error. RL load	124
4.22. Current relative error zoom. RL load	124
4.23. Active power relative error. RL load	125
4.24. Active power relative error zoom. RL load	126
4.25. Reactive power relative error. RL load	126
4.26. Reactive power relative error zoom. RL load	127
4.27. SM_computations subsystem used for simulation with inverter	128
4.28. Grid tied inverter	129
4.29. Current control scheme	129

4.30. Simplified diagram of the simulation	130
4.31. RMS voltage. Inverter load	131
4.32. Voltage relative error. Inverter load	132
4.33. RMS current. Inverter load	132
4.34. Current relative error. Inverter load	133
4.35. Current relative error zoom. Inverter load	133
4.36. Active power. Inverter load	134
4.37. Active power relative error. Inverter load	134
4.38. Active power relative error zoom. Inverter load	135
4.39. Reactive power. Inverter load	135
4.40. Reactive power relative error. Inverter load	136
4.41. Reactive power relative error zoom. Inverter load	136
4.42. Model of a feedback system	137
4.43. Identification scenario of an active HUT	141
5.1. Algebraic loop example	146
5.2. Subsystems at deadlock	147
5.3. The subsystems executing in series	147
5.4. Subsystems executing in parallel	147
5.5. Block handling I/O	149
5.6. Detail of I/O block	150
5.7. Real time simulator internal architecture.	151
5.8. Operational limits of Amplifier	152
5.9. Amplifier bandwidth	153
5.10. Amplifier step response	154

5.11. Delta electronika DC source	154
5.12. Huawei inverter topology	156
5.13. Experiment setup with passive load	157
5.14. Experiment with passive load: RMS Voltage.	158
5.15. Experiment with passive load: % of error for RMS voltage.	159
5.16. Experiment with passive load: % of error for RMS voltage. Zoom	159
5.17. Experiment with passive load: RMS current.	160
5.18. Experiment with passive load: % of error for RMS current.	160
5.19. Experiment with passive load: % of error for RMS current. Zoom	161
5.20. Experiment with passive load: Active power.	161
5.21. Experiment with passive load: Active power. % of error for active power.	162
5.22. Experiment with passive load: % of error for active power. Zoom.	162
5.23. Experiment with passive load: Reactive power.	163
5.24. Experiment with passive load: % of error for reactive power.	163
5.25. Experiment with passive load: % of error for reactive power. Zoom.	164
5.26. Experiment setup with solar inverter	165
5.27. RMS voltage during test	166
5.28. RMS current during test	166
5.29. Active power during test	167
5.30. Reactive power during test	167
5.31. Active power during test. Zoom	169
5.32. Reactive power during test. Zoom	169
5.33. Current waveform before first Z_{DIM} update while feedback is on	170
5.34. Current waveform after first Z_{DIM} update while feedback is on	170

5.35. Current waveform during first Z_{DIM} update while feedback is on	171
5.36. RMS voltage during second test	172
5.37. RMS current during second test	173
5.38. Active power during second test	173
5.39. Reactive power during second test.	174
5.40. Active power during second test. Zoom	174
5.41. Reactive power second during test. Zoom	175
5.42. Current waveform with almost no phase error	175
A.1. Huawei inverter technical data	190
A.2. Puissance Plus power specification	191
A.3. Puissance Plus accuracy specification	191
A.4. Puissance Plus time specification	192
A.5. Puissance Plus physical specifications	192
A.6. OP1400 amplifier technical data	193

TABLE INDEX

1.1. Amplifier comparison	9
2.1. System Parameters	26
2.2. Stability Margins. ITM	27
2.3. Stability Margins. ITM with low pass filter - System 1	33
2.4. Stability Margins. PCD - System 1	41
2.5. Stability Margins. DIM - System 1	55
2.6. Interface algorithms: Summary	70
2.7. Interface algorithms: Open-loop transfer functions	70
3.1. Parameters of identified systems	90
4.1. Simulation parameters: passive load	117
4.2. Simulation parameters: Inverter load	130
5.1. Experiment parameters: passive load	157
5.2. Experiment parameters: inverter load	165

RESUMEN

El testeo y la validación son partes esenciales del desarrollo de cualquier dispositivo eléctrico de potencia. Las simulaciones Power Hardware in-the-loop pueden desempeñar un papel vital en la mejora de la fase de prueba de estos dispositivos, permitiendo escenarios de prueba realistas y, reducir costes, riesgos y ahorrar tiempo. Esta tesis presenta un trabajo de investigación sobre la implementación práctica de una plataforma de simulación PHIL utilizando el Damping Impedance Method. Para tener éxito, esto requiere la identificación de la impedancia del dispositivo bajo prueba. Presentamos una estrategia de identificación basada en el análisis espectral, para obtener una identificación no paramétrica de la impedancia del HUT, seguida de una rutina de ajuste que obtiene una representación paramétrica de la misma. A continuación se simula la plataforma PHIL para probar su funcionamiento así como el de la rutina de identificación con HUT pasivos y activos. La implementación de la plataforma PHIL se realiza en un simulador multi-núcleo de tiempo real, lo que permite que la simulación eléctrica y la rutina de identificación se ejecuten en paralelo, en un mismo dispositivo. Se realiza luego, una verificación experimental de los escenarios simulados. Se encuentra una diferencia fundamental entre la identificación de los HUT pasivos y activos y se proponen y aplican estrategias sobre cómo hacer frente a ella. Finalmente, se dan recomendaciones sobre la realización de más simulaciones PHIL con la plataforma.

Palabras Claves: Power hardware in-the-loop, Damping impedance method, Solar inverter, Interface algorithm, Real-time, OPAL-RT

ABSTRACT

Testing and validation are essential parts of the development of any electrical power device. Power hardware in-the-loop simulations can play a vital role in improving the testing phase of these devices, allowing for realistic testing scenarios while at the same time reducing costs, risks and saving time. This thesis presents research on the practical implementation of a PHIL simulation platform using the damping impedance method. To be successful this requires the identification of the impedance of the hardware under test. We present an identification strategy based on spectral analysis, to obtain a non-parametric identification of the HUT's impedance, followed by a fitting routine that obtains a parametric representation of it. The PHIL platform is then simulated to test the identification routine with passive and active HUTs. Implementation of the PHIL platform is done in a multi-core real-time target, allowing for the electrical simulation and identification routine to run in parallel, on the same device. Experimental verification of the simulated scenarios is performed. A fundamental difference between the identification of passive and active HUTs is found and strategies on how to cope with it are proposed and implemented. Finally, recommendations are given about the realisation of further PHIL simulations with the platform.

Keywords: thesis template, document writing, (Write here the keywords relevant and strictly related to the topic of the thesis).

1. INTRODUCTION

Testing and validation are essential parts of the development cycle of any complex engineering project. Whether this project is structured using a V-model (Fowler, 2015), a waterfall model (Petersen, Wohlin, y Baca, 2009), an iterative model (Grogan, De Weck, Ross, y Rhodes, 2015), or any of the many popular project development methodologies, testing and validation will always take up a significant part of the system's development.

Power electrical devices are no different, and are often subjected to rigorous and extensive testing during their development. Furthermore, countries and international organisations often dictate norms and standards that these devices must comply with, in order to be introduced into the market. This is specially true for grid connected devices, where to be allowed interconnect with the electrical grid, they must comply with a series of standards and regulations covering subjects such as safety (Parise, 2013), operational capabilities (IEEE Standard Association, 2018), compliance with communication standards (IEEE Standard Association, 2011), (IEC, 2010), among many others. Devices such as synchronous machines, wind turbines, residential solar inverters, battery storage devices or a simple transformer, must all undergo lengthy testing and validation before being ready to enter the market and be put in service. Even devices which are not required to meet a certain standard still go through lengthy testing procedures as engineers seek to ensure they perform accordingly to the expectations set during the design stage. This may prove challenging, as it often involves the design of complex testing environments meant to realistically replicate the conditions to which a device will be exposed during its operation.

Whichever the case, Power Hardware in-the-loop (PHIL) simulations can play an invaluable role during the testing phase of a product's development, potentially reducing costs, saving time, reducing risks, and improving the safety and accuracy of the testing procedures.

1.1. The role of power hardware in-the-loop in testing

The approaches used for testing power electrical devices are many, ranging from off-line simulation using simplified models, to physical testing done with the finished product. We can organise these different approaches as seen in the diagram below 1.1.

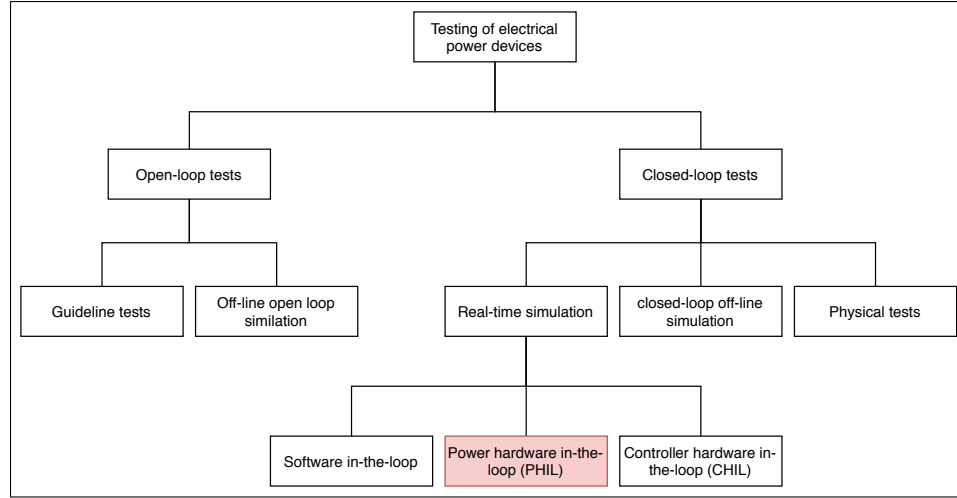


FIGURE 1.1. Testing methodologies

We first, divide the different methodologies in either open-loop or closed-loop testing. This distinction related to the ability of the device tested, or Hardware Under Test (HUT), to exert influence to the testing environment. In open-loop tests, the testing conditions are pre-set at the start of the test and the actions of the HUT are not able to influence them; whether because the HUT is incapable or because these influences are simply not fed-back to generate the appropriate change.

Open-loop tests are often used to certify compliance with a standard. These type of tests can be called guideline tests, as the tests variables applied to the HUT must follow a specific set of predefined guidelines in order for the test to be valid.

An example of this type of test would be the one mandated by the IEEE 1547.1 standard to verify the voltage support features for distributed energy resources (DERs). In this test, a source varies the RMS voltage level at the point of interconnection (POI) with the

HUT. A voltage profile is preset for the test, and to pass, the HUT must perform a set of acceptable actions in response to that profile. Figure 1.2, taken from (IEEE Standard Association, 2020), shows an example of a Low voltage ride-through (LVRT) test profile.

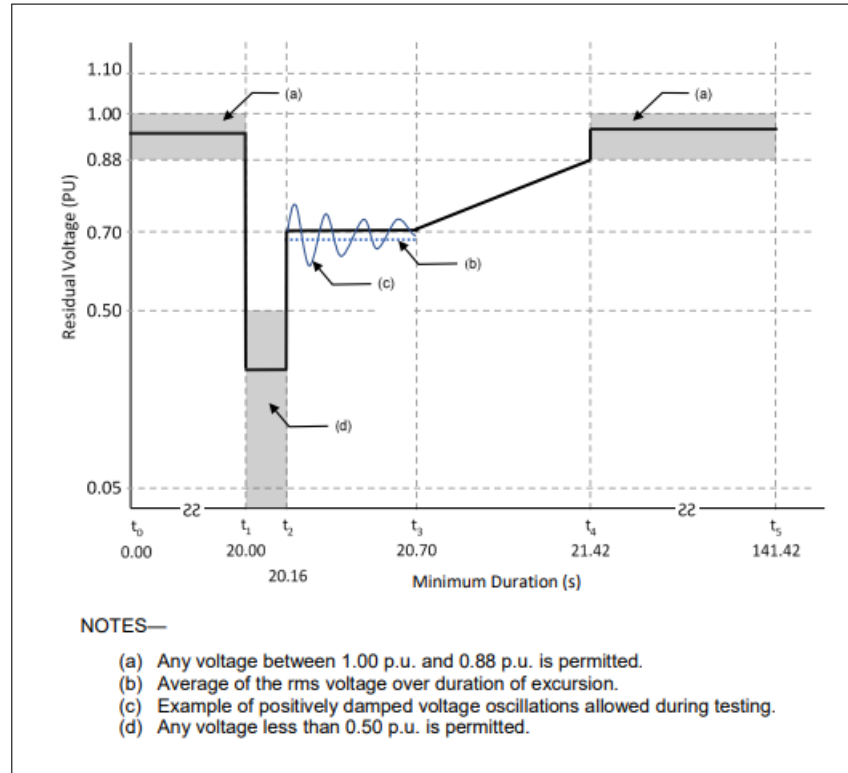


FIGURE 1.2. LVRT voltage profile. [2020] IEEE

During this test it is explicitly required that the actions of the HUT in no way affect the voltage profile applied to it by more than 1 % of the set value. While this may be reasonable since the goal of the test is grid compatibility; it must be noted that this behaviour is not necessarily an accurate representation of the conditions the HUT will encounter during operation. One would expect the HUT to have no impact on the voltage of its POI, if this has a very low short circuit ratio (ie: the HUT is connected to a very strong grid). In reality though, it is easy to think of common scenarios where this would not be true, such as the HUT begin connected at the end of a distribution feeder or in a temporarily islanded microgrid. Here, we would expect the response of the HUT to significantly affect

the voltage of its POI. The change in the voltage produced by the device, will in turn have an affect how the device responds and so on, possibly leading to the HUT presenting a very different response than that of the standard test. We can see that while useful, these types of tests may not be enough to ensure the adequate operation of a device under real conditions.

By contrast, closed-loop tests take into consideration the effects the operation of the HUT has, and alter the testing conditions accordingly. Because of this, they can provide a much better approximation of the behaviour of the HUT in a given situation.

The simplest example of this would be a complete physical test or a field test. In this type of test, any action the HUT takes, has an effect in the rest of the system and vice versa, simply because of the physics of the situation. It goes without saying that this test would provide the results closest to reality, because it is reality. However, it is also easy to see that this method of testing is extremely expensive as each scenario involves the construction of a physical system to test the HUT. An example of this type of testing facility is the REIDS project off the coast of Singapore. A microgrid testbed built in collaboration between Engie, Schneider Electric and the Nanyang Technological University ([Peng y Wild, 2017](#)).

Today, the most popular form of testing, by far, is the use of computer simulation. Just like physical testing, circuit simulation is also a closed-loop method as the effects the HUT and the rest of the simulation have over each other are calculated and applied at each time-step. Powerful circuit simulators such as Power Factory, Matlab/Simulink or PLECS have made possible to quickly and cost-effectively develop models of electric devices. These models can be easily modified and tested in complex simulated environments that are quick to build and adjust. All these advantages have made simulation the primary tool used to develop and test electrical devices. The main problem with computer simulation is that the fidelity of the results is dependant on how accurately the models represent the real device. Provided with precise models, simulation can be amazingly accurate but highly

precise models are often difficult to produce. The effects of parasitic components, electromagnetic interference, computational delays of micro-controllers and communication problems may all cause the real device to behave differently from the simulated model, under the right conditions. Because of this, even the most confident developers still perform extensive hardware tests to validate their results and contrast the dynamics of the real device with the models used to develop it.

It is here where real-time simulation techniques such as Controller Hardware in-the-loop (CHIL) and Power hardware in-the-loop (PHIL) can significantly contribute to the testing of an electrical device. These two techniques aim to combine the best elements of physical testing and computer simulation. Figure 1.3 shows the typical configurations used in CHIL and PHIL experiments.

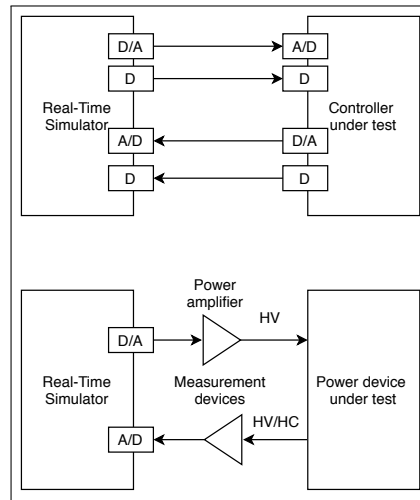


FIGURE 1.3. CHIL and PHIL architecture

In a CHIL experiment the HUT is the real physical device, often a microcontroller designed to control some system. This device is interfaced with a simulated testing environment that runs in real-time. This means that time in the simulation runs at the same speed as in the outside world. This allows the HUT to react to the stimuli coming from

the simulation and respond to it as it would under real operating conditions. The simulation in turn, senses the response of the HUT, calculates its effect on the simulated system and modifies the variables presented to the HUT. The communication between the HUT and the simulation can be done through analog and/or digital signals, depending on the requirements of each specific experiment.

This testing methodology is extensively used in the automotive industry. Here, the independent systems of the vehicles are connected to a simulation of the rest of the car. This allows the development process to advance at a much faster pace as, for example, the team that develops the throttle system does not need the motor and transmission to be ready, but only a simulation of them, to implement and test their system on the micro-controllers that will be used in the car. This approach allows for the fidelity of physical testing and the ease of implementation and flexibility of computer simulation.

Power hardware in-the-loop is the extension of this idea, to power devices and electrical networks. In a PHIL experiment the HUT is a power device interfaced to a simulated electrical system through the use of a voltage or current amplifier. Like in CHIL experiments, the electrical system is simulated in real time and the effects of the HUT, measured and incorporated to it at every timestep. The amplifier then applies the calculated voltage or current to the HUT, and so on, completing the feedback loop. This approach retains many of the benefits of CHIL experiments, as fidelity and flexibility will be high.

PHIL simulations are specially suited for experiments where the effects of the HUT on the testing environment are predicted to be large, or are also of interest. An example of this would be, like presented before, the simulation of a distribution feeder where a relatively large DER is connected. Here, the effects of the network on the HUT would likely be as important as the effects the HUT has on the network. This problem in particular, has received much interest by researchers and industry alike, with the advent of distributed generation (Radatz, Kagan, Rocha, Smith, y Dugan, 2016), (Nassif, 2018), (Wang y De Leon, 2020).

Another example where PHIL would be particularly useful, would be the in testing of motor drives. Here, a simulated motor could be connected to the real drive (Lemaire, Sicard, y Belanger, 2015), (Zyuzev, Mudrov, y Nesterov, 2016). We can see how simulating the responses of the motor to the control actions of the drive, is of crucial importance to asses its performance. A PHIL simulation allows to perform these types of tests, even if the motor is not yet built.

Nonetheless, PHIL simulations pose additional challenges when compared to CHIL, often making them more difficult to implement. Instability may occur as a consequence of the properties of the interface used to connect both systems. Also, the non-idealities of this interface may also affect the accuracy of the simulation (Ren, Steurer, y Woodruff, 2007). Thus, while PHIL simulations present a great opportunity to improve and speed-up the testing process of a power device, they must be implemented taking the appropriate considerations, to achieve good results.

1.2. State-of-art of PHIL

As explained in the previous section, Power Hardware in-the-loop (PHIL) is a real-time simulation technique in which a physical power device, often referred to as *Hardware Under Test* (HUT), and a virtual electrical system, referred to as *Rest of System* (ROS), are connected via an interface allowing them to exchange power. Physically this is achieved by power amplifier able to supply or sink the power dictated by the electrical simulation. Computationally this interface is performed by an *Interface Algorithm* (IA) which is responsible for commanding the power amplifier and feeding back the effects of the real system into the simulation. Both the amplifier and these interface algorithms, are key to the success of a PHIL experiment and have been subject of much study in literature.

For the amplifier, two main technologies exist: switch-mode amplifiers and linear amplifiers. Switching amplifiers are usually AC/DC/AC power converters (they can be AC/DC for DC PHIL experiments) that use a variety of control and modulation strategies

to track the reference given by the real-time simulation (Karapanos, De Haan, y Zwetsloot, 2011). To simulate a 3-phase system, 3-leg and 4-leg inverters are used, the latter to allow for unbalanced operation. Multiple control strategies have also been suggested and implemented, such as: PI, PID, predictive control, etc. Different topologies for filters have also been suggested in the literature, but the most common remain the LC and LCL filters (Lehfuss et al., 2012). Switch-mode amplifiers are convenient for PHIL applications due to their relatively low cost per kW of power and their ability to operate in 4 quadrants without much restrictions, as they can just feed power back to the grid. They are also highly efficient and available in a wide range of power, from just a few kW to several MW. They can be implemented in a modular fashion and are often the only option for high voltage applications. Their main drawback is their generally high input lag and low bandwidth giving them a comparatively poor dynamic performance. Moreover, their complex structure and dynamics can have a significant effect in the stability of a PHIL simulation, often in unintuitive ways (Marks, Kong, y Birt, 2018b). Some attempts of using high switching frequencies have succeeded on mitigating these drawbacks (Benigni, Helmedag, Abdalrahman, Piłatowicz, y Monti, 2011).

The second technology used for PHIL amplification are linear amplifiers. These amplifiers use an amplification stage consisting on multiple MOSFETs in parallel operating in their linear region. The control of this type of amplifier is very simple, as the reference can often just be fed to the amplification stage directly. However, many devices add a compensation loop that performs load regulation. The main benefit of this type of amplifier is its performance. Input delay is much lower than switching amplifiers of comparable power and bandwidth is also higher. Harmonic distortion is also very low, as there is no switching activity going on. This makes them very well suited for simulations where accuracy is key. Yet, linear amplifiers have many drawbacks when compared with their switch-mode counterparts. They are more expensive in a per kW basis, they are less efficient as the MOSFETs operate in their linear region, and they have trouble sinking their rated power, as they normally cannot feed power back to the grid and rely on internal

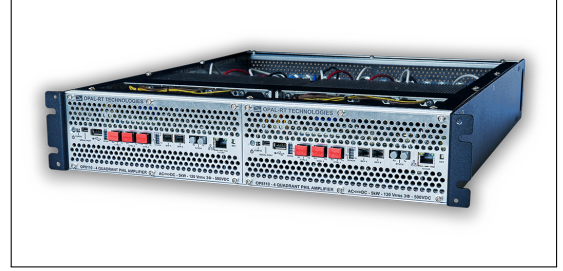
resistors and fans to dissipate the excess power. This in turn makes them less compact, requiring more valuable lab space. They cannot handle high voltages as their design makes it so, the MOSFETs must bear a large part of the rated voltage of the device. This means that linear amplifiers are confined to low to mid power applications. A comparison between two commercial PHIL amplifiers can be seen in table 1.1, figures 1.4a, 1.4b and further technical information about both devices can be found in annex A. While PHIL amplifier design and control is a fascinating topic, worthy of much discussion, it is outside the scope of this thesis.

TABLE 1.1. Amplifier comparison

Device	PA-3X7000-AC-DC-4Q-400V	OP1400 Series
Brand	Puissance Plus	OPAL-RT
Technology	Linear amplifier	Switching amplifier
Semiconductors	Silicon MOSFET	SiC MOSFET
Power Range	21kW	10-15kW
Voltage range	0-565V RMS / ± 1120 V DC	0-380V RMS / ± 400 V DC
Phases	3 phases	3, 6 or 9 phases
Bandwidth (-3db)	DC - 70kHz	DC - 10kHz
Slew rate	$65V/\mu s$	$5V/\mu s$
Input delay	$< 7\mu s$	$5,5\mu s - 8,3\mu s$
THD	$< 0,7\%$ (max)	$< 2\%$ @10kHz
Absorption capacity	50 % (dissipative)	100 % (non-dissipative)
Size	1950mm x 800mm x 800mm (38U)	< 178 mm x 482.6mm x 500mm (4U)



(A) Puissance Plus PA-3X7000-AC-DC-4Q-400V amplifier



(B) OP1400 amplifier

FIGURE 1.4. Amplifier size comparison

The second part of a PHIL interface is the interface algorithm. The early successful implementations of PHIL used the simplest IA of all, the *Ideal Transformer Method* or ITM, see in figure 1.5. This method consists of a controlled voltage source in the hardware side and a current source that feeds back the current from the HUT to the simulation. We see the ITM being used in applications such as a test bed for electric ship motors [Liu et al. \(2005\)](#), [Steurer et al. \(2007\)](#), and for PV converters [Seo et al. \(2011\)](#) and [Langston et al. \(2012\)](#). However, researchers quickly realised the downsides of the ITM. Its poor stability made it so applications often required the inclusion of filters on the feedback current, decreasing accuracy ([Iwado, Ohori, Hattori, y Funaki, 2015](#)). To alleviate this problem, several researchers applied compensation strategies to the ITM. In [Ainsworth et al. \(2016\)](#) we see how these compensation techniques are applied in an experiment where a battery inverter is connected to a simulated distribution feeder. In the experiment, a simulated transformer and the large and slow power amplifier introduce notable distortions in the simulation. In [Marks et al. \(2018a\)](#) we see a more general approach for compensation of ITM interfaces.

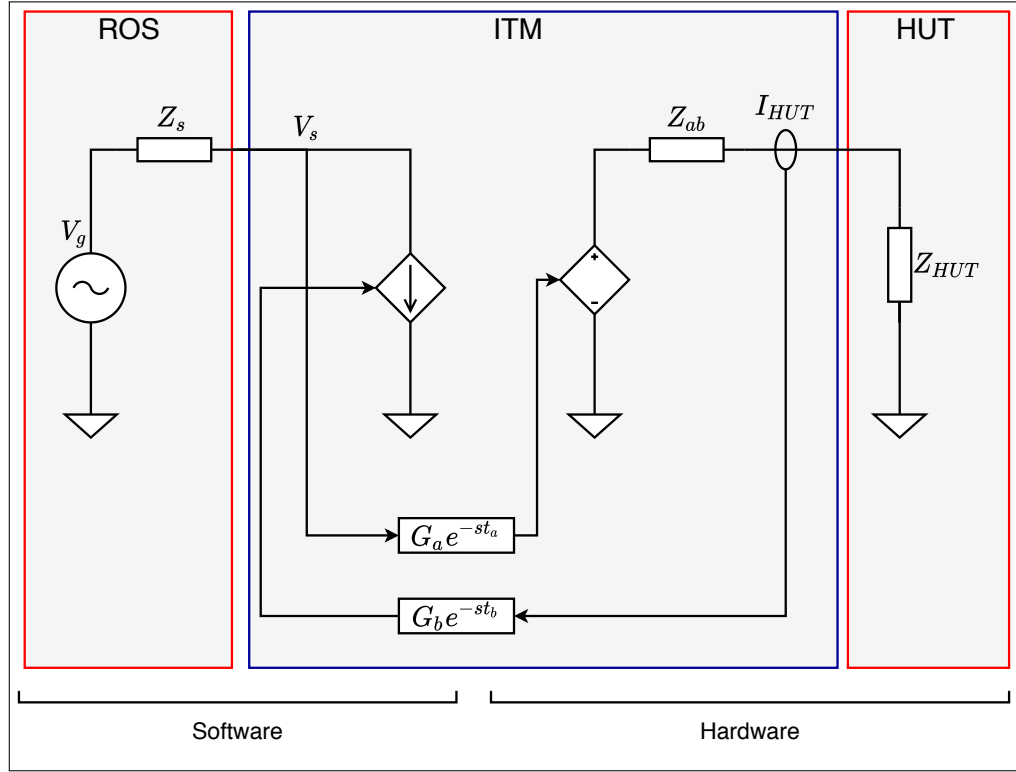


FIGURE 1.5. ITM interface algorithm

Over the years, other interface algorithms were proposed such as the Transient First-order Approximation (TFA) (Wu, Lentijo, y Monti, 2004), Transmission line model (TLM) (Wu y Monti, 2005), Multi-rate partitioning (Lehfuß, Lauss, y Strasser, 2012), etc. But none have shown considerable improvements in stability and performance over the ITM with compensation and are not widely used.

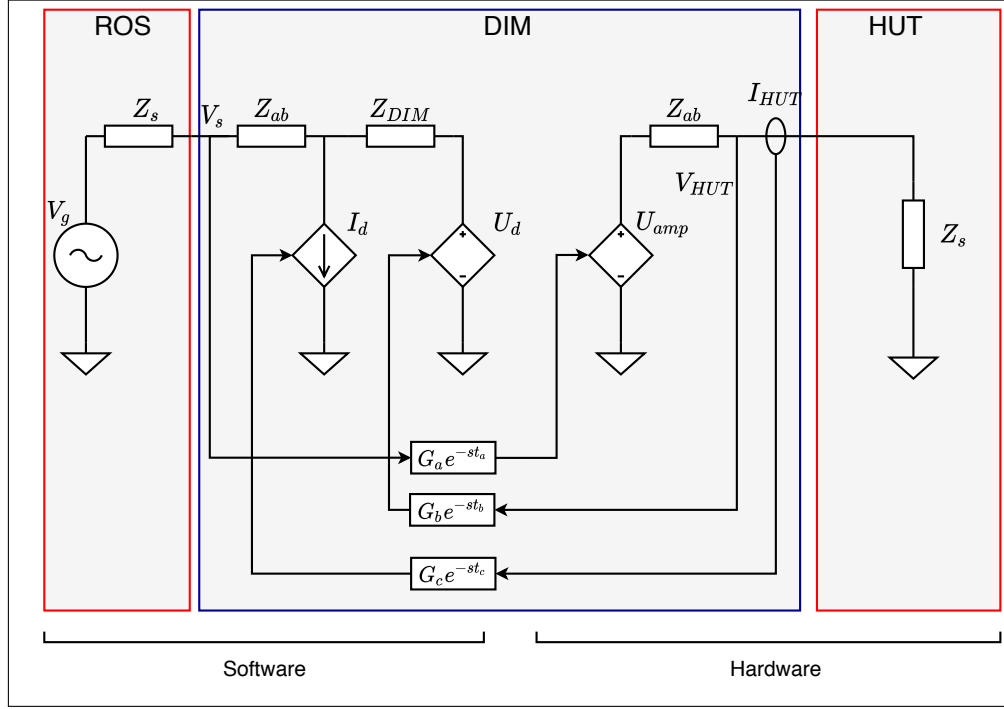


FIGURE 1.6. DIM algorithm

An interface algorithm that has shown significant improvements in both stability and accuracy over the ITM is the *Damping Impedance Method* or DIM (Brandl, 2017) and (Hatakeyama, Riccobono, y Monti, 2016). This algorithm differs from the ITM as it adds a virtual damping impedance Z_{DIM} in the software side. It also feeds back both the current and voltage of the HUT. A diagram of the DIM IA can be seen in figure 1.6. The main advantage of the DIM algorithm is that its properties depend on the choice of the Z_{DIM} . It has been shown that if this damping impedance perfectly matches the impedance of the HUT, the simulation is guaranteed to be stable (Zhang, 2016). Further more, accuracy is also preserved. This raises a new challenge though, as knowledge of the HUT's impedance is unlikely to be available. Because of this, several implementations of the DIM algorithm use some method to estimate Z_{HUT} .

Some implementations of PHIL platforms using the DIM IA use very simple methodologies to estimate the Z_{HUT} . In Paran y Edrington (2013) an estimate is obtained by

using the magnitudes and phase of the voltage and current. It is assumed that the HUT can be modelled as an LR filter and parameters for R and L are obtained, as seen in equations 1.1 through 1.3.

$$\frac{RMS(V)}{RMS(I)}(\cos\varphi + j\sin\varphi) = R_{DIM} + j\omega L_{DIM} \quad (1.1)$$

$$\frac{RMS(V)}{RMS(I)}(\cos\varphi) = R_{DIM} \quad (1.2)$$

$$\frac{RMS(V)}{RMS(I)}(\sin\varphi) = \omega L_{DIM} \rightarrow \frac{RMS(V)}{RMS(I)} \frac{\sin\varphi}{\omega} = L_{DIM} \quad (1.3)$$

This approach has been applied by [Paran, Fleming, Li, y Edrington \(2014\)](#) in simulations with harmonics loads and by [Jiang, Li, Xin, Wang, y Wang \(2019\)](#) in a PHIL simulations of MMC-HVDC device. Despite the relative success of this method, there are clear drawbacks to it. First, it is not easy to generalise as different HUTs will have different topologies for Z_{HUT} . Assuming that every HUT can be treated as an LR filter is not accurate to represent a variety of HUTs. Second, the estimate will only prove accurate for the fundamental frequency used to calculate the RMS values and phase, this means that if an HUT presents some other dynamics at higher frequencies, they will not be taken into consideration, reducing the accuracy of the simulation.

A second approach to estimating Z_{HUT} has been proposed by [Siegers y Santi \(2014\)](#) and implemented by [Liegmann et al. \(2016\)](#) and [Riccobono et al. \(2017\)](#). This approach uses a technique called Wide band Identification to obtain an estimate of the impedance of the HUT across a large range of frequencies. This is essentially a spectral analysis technique where correlation analysis is used to obtain a non-parametric identification of a system. This information is then used to derive a parametric model for Z_{HUT} . The method proposed by the authors, makes up for the shortcomings of the previous attempts to use the DIM IA, by obtaining a much more accurate estimate of Z_{HUT} without needing prior knowledge of its topology. This improvement however, came at the cost of heavily increasing the computational burden. To cope with this, the author resorted to running the

estimation procedure in a dedicated real-time PC, regularly communicating the estimates for Z_{HUT} to the electrical simulator. A diagram of the system implemented by [Liegmann et al. \(2016\)](#) and [Riccobono et al. \(2017\)](#) is shown in figures 1.7 and 1.8.

Measurements of the current and voltage of the interface are taken and used by the dedicated real time PC using LabVIEW. This PC calculates the Discrete Fourier Transform of both measurements and uses it to obtain a non parametric estimate of the impedance. Then it uses a fitting routine based on the least squares method to the a parametric transfer function. Finally, the known topology of the Load is used to determine parameters for R, L and C. The detailed process used is shown in figure 1.8. This estimate for $Z(s)$ are then passed to another real-time PC that runs the electrical simulation and controls the amplifier via V_{ref} in figure 1.7. The results obtained with this approach seem to confirm the claims made by the authors of increased accuracy.

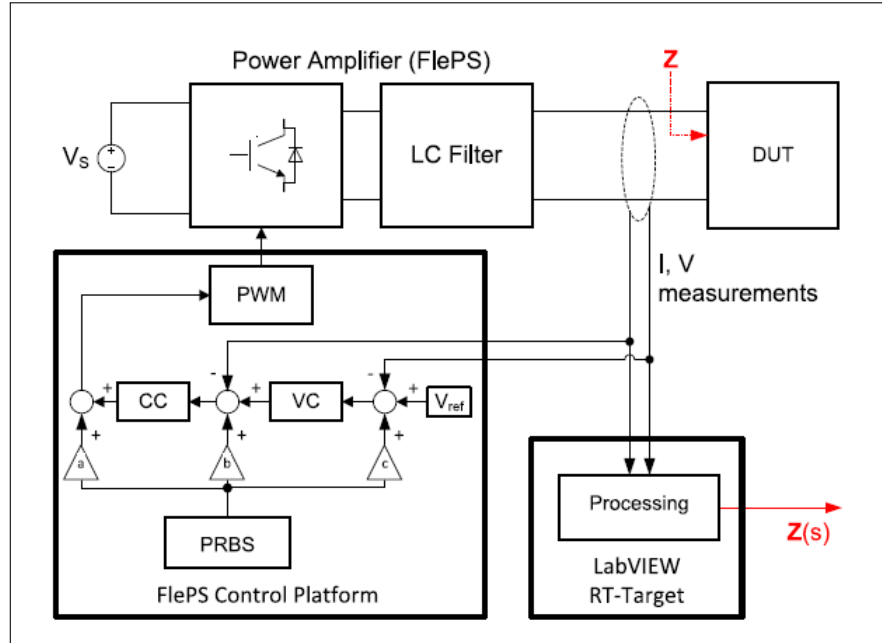


FIGURE 1.7. Block diagram of the WSI technique. [2017] IEEE

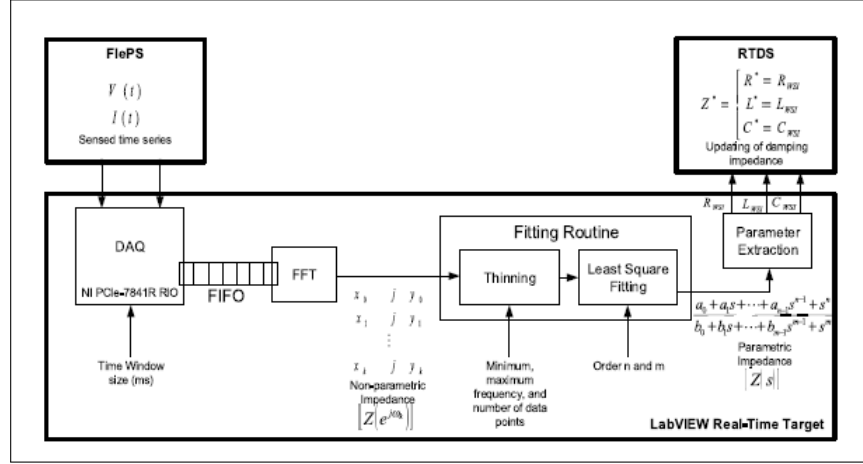


FIGURE 1.8. Block diagram of the postprocessing routine to identify the parametric impedance. Implemented in LabVIEW [2017] IEEE

1.3. Original contribution of the thesis

The main contribution of this thesis is the implementation of a PHIL platform that uses the damping impedance method, as well as an identification routine to allow it to adapt to different HUTs. This thesis was performed in collaboration with Engie Laborelec and the experimental work was performed in their facilities at Linkebeek, Belgium. Their main interest in the implementation of this PHIL platform was that it should be capable of performing the widest possible range of PHIL simulations, utilising varied HUTs. Thus, a thorough analysis of the different interface algorithms was performed in chapter 2 where the strength and weaknesses of each were evaluated, arriving at the conclusion that the DIM IA was the best alternative. Given that the DIM IA was selected, a method for identifying the HUTs impedance was needed. Seeing the success of the approach by [Liegmann et al. \(2016\)](#) and [Riccobono et al. \(2017\)](#), a similar strategy was implemented, described in chapter 3. Novel contributions were made in the implementation of this strategy. Taking advantage of the multi-core nature of the OP5600 real-time target from OPAL-RT, allocation of separate cores for the electrical simulation and the identification procedure was possible, eliminating the need for a second real-time target. This allowed us to run

the electrical simulation and identification routine in parallel without them significantly interfering with one another. This allows for a much simpler setup, requiring only 3 main components: the OP5600 real-time target, a 21kW 4-quadrant linear amplifier by Puisseance Plus, and the HUT. Furthermore, the entire setup can be programmed and modified using only MATLAB/SIMULINK and OPAL-RT's RT LAB software.

We verify the results obtained by [Liegmann et al. \(2016\)](#) and [Riccobono et al. \(2017\)](#) through both simulation and physical experimentation with similar types of passive loads. We also performed both simulation and experimental verification of the application of this technique to active HUTs, in our case a residential solar inverter (Huawei SUN2000L-3KTL). The results allow for a more careful understanding of the strengths and limitations of the strategy used, helping to understand better its possible use cases and further development.

1.3.1. Objectives

The objectives of this thesis are all related to the successful implementation of a flexible PHIL platform, capable of performing stable and accurate PHIL simulations in a wide range of experimental conditions.

1.3.1.1. General objective

To make a comprehensive analysis of the different interface algorithms used in PHIL simulations and select one for implementation. Then, the implementation of flexible PHIL simulation based in DIM impedance identification techniques, capable of a wide range of simulations with different HUTs.

1.3.1.2. Specific objectives

1. Review the properties of the different interface algorithms.
2. Implement a PHIL platform.
3. Test the PHIL platform with passive loads to verify the results obtained in other works and through simulation.

4. Test the PHIL platform with active HUTs and evaluate its performance.
5. Provide a comprehensive analysis and recommendations towards the realisation of PHIL experiments in the future, using the platform.

1.3.2. Hypothesis

The formulation of the hypothesis is in relation to the implementation of a functioning PHIL platform.

H₁ Among the different interface algorithm, the damping impedance has the best performance in terms of stability and accuracy.

H₂ The strategy selected to estimate the impedance of the HUT can provide accurate estimates for passive and active HUTs.

1.3.3. Methodology

To meet the objectives and prove or disprove the hypothesis, first we will perform a theoretical analysis of the different IAs and study their properties. We will then perform simulations to confirm the predictions of the theoretical analysis. Finally, the PHIL platform is implemented and the simulated scenarios are replicated to confirm or refute our predictions.

1.3.3.1. Evaluation methodology

To evaluate the stability and accuracy of the different interface algorithms we will select two systems with different ratios between the software impedance and the hardware impedance, and then compare the effects each interface algorithm has on the systems through the use of nyquist and bode plots. The accuracy will be evaluated in relation to the *Naturally coupled system* or NCS, the system where no interface is present.

During the simulations, the same approach will be taken. The system without the PHIL interface will be simulated, to provide a baseline. The results of the simulations of the PHIL platform will be compared to this baseline.

The same approach will be taken during the experimental phase, in the case of the passive load. The results of the experiments will be compared with the baseline obtained through simulation. Finally for the active HUT, as its topology and control structures are unknown we can only evaluate its performance qualitatively. To determine if the simulations is accurate or not, we will observe whether the measured variables display the same characteristic behaviour present in all other accurate PHIL simulations.

1.3.3.2. Evaluation variables

To evaluate the stability of the different IAs quantitatively we will compare:

1. Gain margin
2. Phase margin

In terms of accuracy, we will compare how well their bode plots, track that of the NCS. This will be done in both absolute terms, and terms of the error between them, measured in dB.

For the simulations, 4 variables will be measured at both sides of the interface. These 4 variables will be compared against the values obtained in the simulation of the NCS.

1. Voltage
2. Current
3. Active power
4. Reactive power

Finally, the same variables will be measured during the practical experiments performed in with the PHIL platform. In the experiments with a passive load, these quantities will be compared against a baseline obtained through simulation. In the case of the active HUT, we will only assess the qualitative behaviour of these variables, given that a baseline cannot be obtained.

1.3.4. Organisation of the thesis

Chapter 2 studies the properties of several interface algorithms in terms of their stability and accuracy. We show how several key variables affect the performance of each interface algorithm and how they may lead to instability or inaccuracies during experiments. We establish which of the interface algorithms presents the best performance and select it for our PHIL platform.

In **Chapter 3** we study the identification routine that will be used to obtain an estimate of the impedance of the HUT, needed for the damping impedance method IA. We explain the theoretical background and mathematical tools needed to perform the non-parametric identification of the HUT. We then explain how we derive from this information a parametric representation of the HUT's impedance, by applying the Levenberg-Marquardt algorithm to fit a rational.

Chapter 4 covers the software implementation of the system in MATLAB/SIMULINK and the results obtained through simulation. We show how the strategy proposed provides good results for passive loads, but that its applications for active devices may be limited.

In **Chapter 5** we present the hardware implementation of the PHIL platform, covering the necessary changes needed to adapt the model for real time operation. We as well present the experimental results that confirm our predictions stemming from the simulations.

Finally, **Chapter 6** contains the conclusions of the thesis, reviewing the main results and providing recommendations for the realisation of PHIL experiments using the platform.

2. INTERFACE ALGORITHMS

2.1. Introduction

In order to link both the real and virtual parts of an electrical system in a PHIL simulation, an interface algorithm (IA) is required. This is the layer of software that ensures the electrical variables seen in the real circuit match those of the simulation.

Many of these interface algorithms trace their origin back to techniques used in circuit simulation software, meant to solve large and complex circuits efficiently. These techniques, called relaxation techniques, split a large circuit in pieces and then link them together by placing interfaces between them. They then solve each sub-circuit separately and calculate their effect on the surrounding circuits. These effects are then used to recalculate each sub-circuit until both sides of a given interface converge (Newton y Sangiovanni-Vincentelli, 1984). Interface algorithms like the ideal transformer method (ITM), partial circuit duplication (PDC) and the damping impedacence method (DIM), are merely variations of these interfaces adapted for real time operation.

However, this adaptation is not without drawbacks. While in an offline simulation one has the luxury of time to perform as many iterations as needed to get the sides of the interfaces to converge, in PHIL simulations that is not the case. This can cause both stability and accuracy problems that can render the results of a given PHIL simulation useless. As such, knowledge of the properties of each IA is key. The exploration of such properties will be the main subject of this chapter.

Ease of implementation is also a major concern. Given the real time nature of PHIL simulations, an interface algorithm that is too computationally demanding may be hard to implement, or require the simulation to have a large time-step, compromising accuracy. Another algorithm may require an extra physical component added to the HUT in order to work, which can lead to a drop in simulation accuracy, added complexity of the simulation setup, or even to large power losses in high power simulations.

With this in mind, several interface algorithms have been proposed (Wu et al., 2004), (Ren, 2007), (Lehfuß et al., 2012). Comparisons and analysis between them are also abundant in the literature (Hatakeyama et al., 2016), (Brandl, 2017).

In this section we proceed to present these different interface algorithms described in the literature, analyse their properties and select one for the implementation of a PHIL simulation platform. The methods covered will be: the *Ideal Transformer Method* or ITM, *Ideal Transformer Method with a lowpass filter*, *Partial Circuit Duplication* or PCD and finally the *Damping impedance method* or DIM. While interesting, other proposed interface algorithms have not proven very effective in experimental setups and thus will not be analysed.

2.2. Ideal transformer method (ITM)

The ideal transformer method is the simplest and most straight forward of the interface algorithms, making it widely used. It is an adaptation of the I-type interface used by circuit solvers, presented in Newton y Sangiovanni-Vincentelli (1984).

Figure 2.1 presents the circuit diagram of the ITM algorithm. Here, the Hardware Under Test (HUT) is represented as impedance Z_{HUT} , highlighted in the rightmost red box. The simulated power system, to which the HUT is connected is represented as its Thevenin equivalent by voltage source V_g and the impedance Z_s , altogether denominated Rest of System (ROS) highlighted on the leftmost red box. Finally the interface algorithm is shown in the middle blue box.

The algorithm consist of one controlled voltage source, an output impedance Z_{ab} representing the power amplifier, and a current source, feeding back the current of the real device into the simulation. Transfer functions G_a and G_b represent the distortions introduced by the amplifier and current sensor respectively. The delay e^{-st_a} is perhaps the most influential quantity as it represents the sum of the computational delay (ie: the timestep of

the simulation) plus the input lag of the amplifier. By comparison e^{-st_b} represents the delay of the current sensor but this quantity is usually much smaller than e^{-sta} , hence much less important.

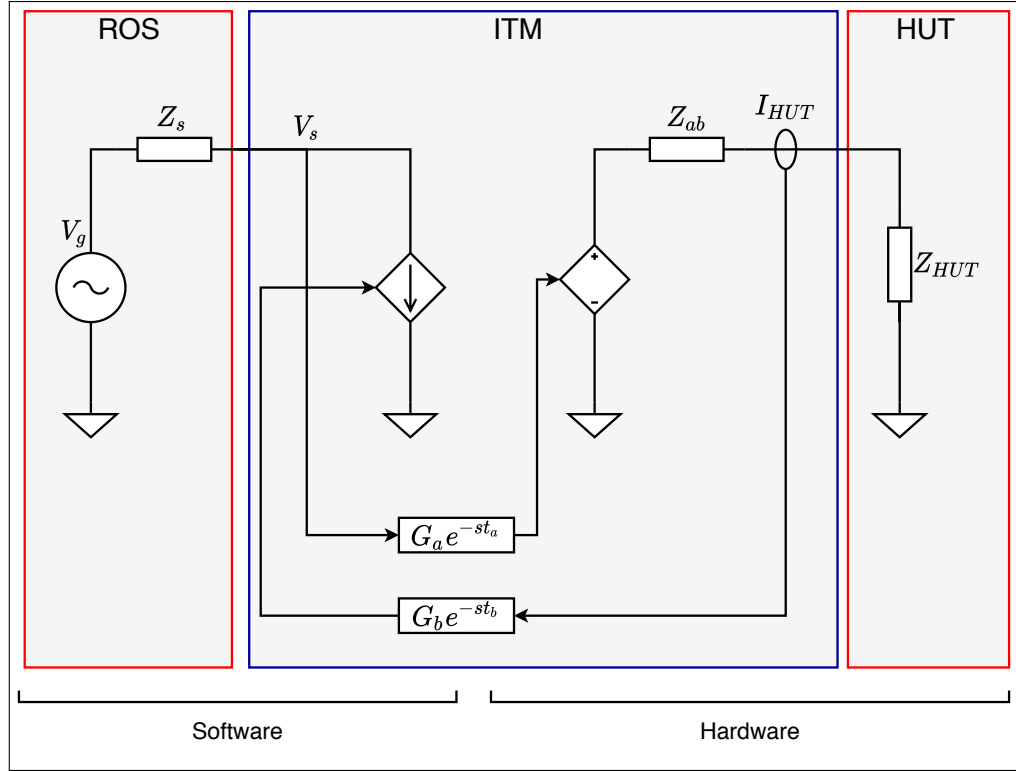


FIGURE 2.1. ITM interface algorithm

2.2.1. Stability considerations

The most important job of an interface algorithm is to deliver a stable PHIL simulation and it is here where the ITM algorithms has its greatest weakness. This can be understood intuitively as explained in [Zhang \(2016\)](#), if we only focus on the two main components Z_s and Z_{HUT} .

Example 2.1. Let $\epsilon[k]$ be a small error in the amplification of V_s to V_{HUT} during timestep k . This error $\epsilon[k]$ would in turn cause an error in the current of the real device.

$$\Delta I_{HUT}[k] = \epsilon[k]/Z_{HUT} \quad (2.1)$$

This current error would then be read by the sensor and fed back into the simulation in the next time step. Maintaining V_g constant, we can see that the amplification error ϵ in timestep k has effectively been amplified by the ratio Z_s/Z_{HUT} as seen in the equation below.

$$V_s[k+1] = V_g - Z_s * I_s[k+1] \rightarrow \Delta V_s[k+1] = -\epsilon[k]Z_s/Z_{HUT} \quad (2.2)$$

From example 2.1, one could theorise that if $\frac{Z_s}{Z_{HUT}} > 1$, an amplification error made at any point would be amplified on each timestep quickly causing the simulation to become unstable and thus, useless. While this is the case when the impedances of both sides are purely resistive, adding an inductive or capacitive component to it makes the analysis more complex. Deriving the open loop transfer function from the block diagram on figure 2.2 can give us a more formal explanation as to why the system can turn unstable.

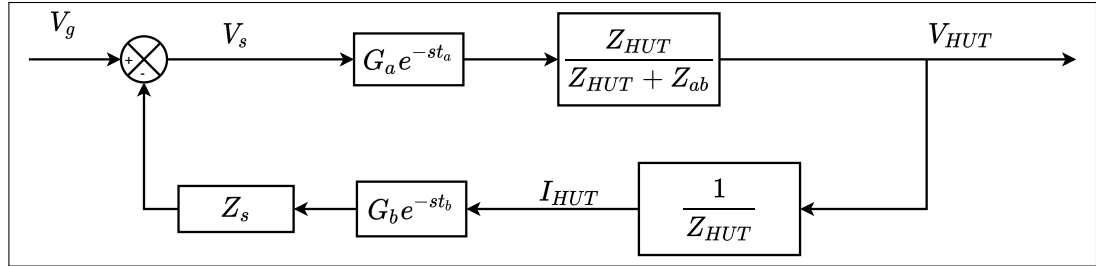


FIGURE 2.2. ITM algorithm block diagram

$$G_{OL} = \frac{Z_s}{Z_{ab} + Z_{HUT}} G_a G_b e^{-s(t_a+t_b)} \quad (2.3)$$

For simplicity, we will ignore the dynamics of the amplifier and sensors, represented by G_a and G_b . These are indeed an important part of the interface algorithm, however, with good component selection their effects on stability should be minimal. This, as the ideal response of these components is to have 0db gain and no impact in the phase, throughout

the relevant frequency range of the experiment. We may also neglect the effects of Z_{ab} , as the output impedance of the amplifier will generally be much smaller than both Z_s and Z_{HUT} .

With this in mind, we can see that the gain of the open loop function will be set by the ratio between Z_s and Z_{HUT} at each specific frequency. At the same time, the delay will cause the phase to drop as frequency increases. This will mean that, if $\|\frac{Z_s}{Z_{HUT}}\| > 1$ when phase reaches -180° the system will enter into positive feedback with a loop gain higher than 0db, becoming unstable.

This has interesting consequences for the case where both Z_s and Z_{HUT} are a resistor and an inductor in series ($Z = R + sL$). Here we see that the gain of the open loop transfer function will be given by the ratio of resistances for low frequencies and by the ratio of inductance at higher end of the spectrum. Thus, a system where $L_{HUT} > L_s$ would always be unstable, regardless of the value of t_a . The opposite case isn't a guarantee of stability either as a system where $L_{HUT} < L_s$, but where $R_{HUT} > R_s$ could also be unstable, if the delay was small enough that the loop gain is still above 0db by the time the phase reaches -180° .

This can be seen in figures 2.3 and 2.4 where we plot the Bode and Nyquist diagrams of two systems where Z_s and Z_{HUT} are modelled as a resistance and an inductance in series. In System 1, the ratio between the software and hardware resistances is 1:2, while the ratio of the inductances is 2:1. This case is unstable, as even if the DC gain is smaller than 1 by the time the delay of the system takes the phase to -180° (highlighted by the dotted red line) the magnitude of the open loop transfer function is mainly determined by the inductance ratio, which is greater than 1. System 2 is the opposite. Here, resistance ratio is 2:1 and inductance ratio is 1:2. Predictably, the system is stable as the delay is small enough, that it only takes the phase to -180 (highlighted by the dotted blue line) after the gain is primarily set by the ratio of inductances. The complete parameters of both

cases can be found in table 2.1. We can quantify the stability of the systems by calculating their respective gain and phase margins, seen in table 2.2.

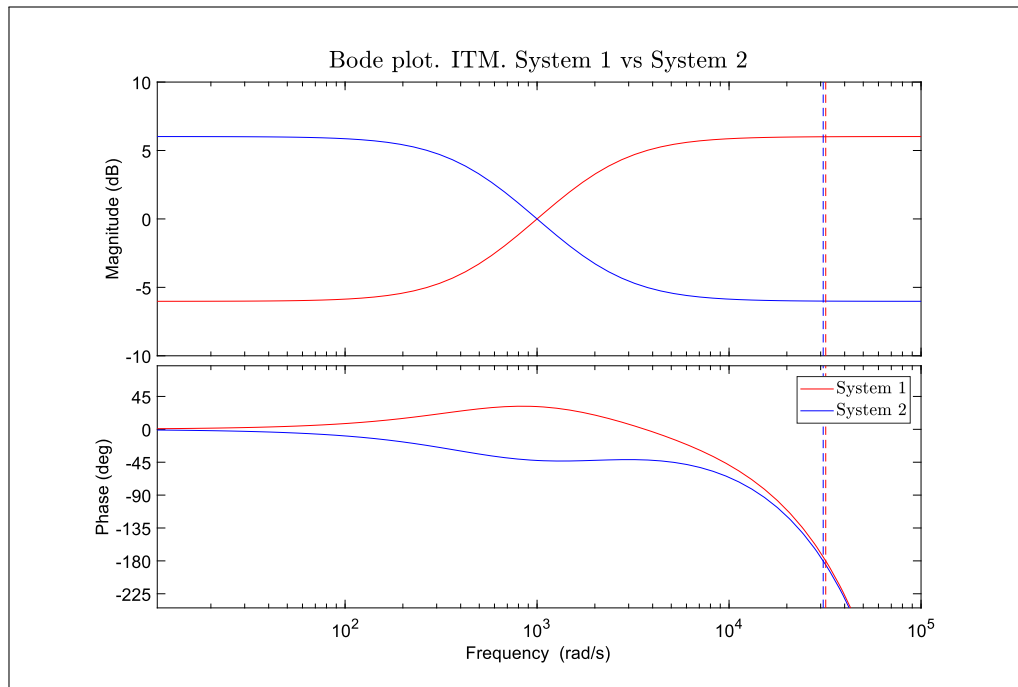


FIGURE 2.3. Comparison between stable and unstable cases of ITM algorithm

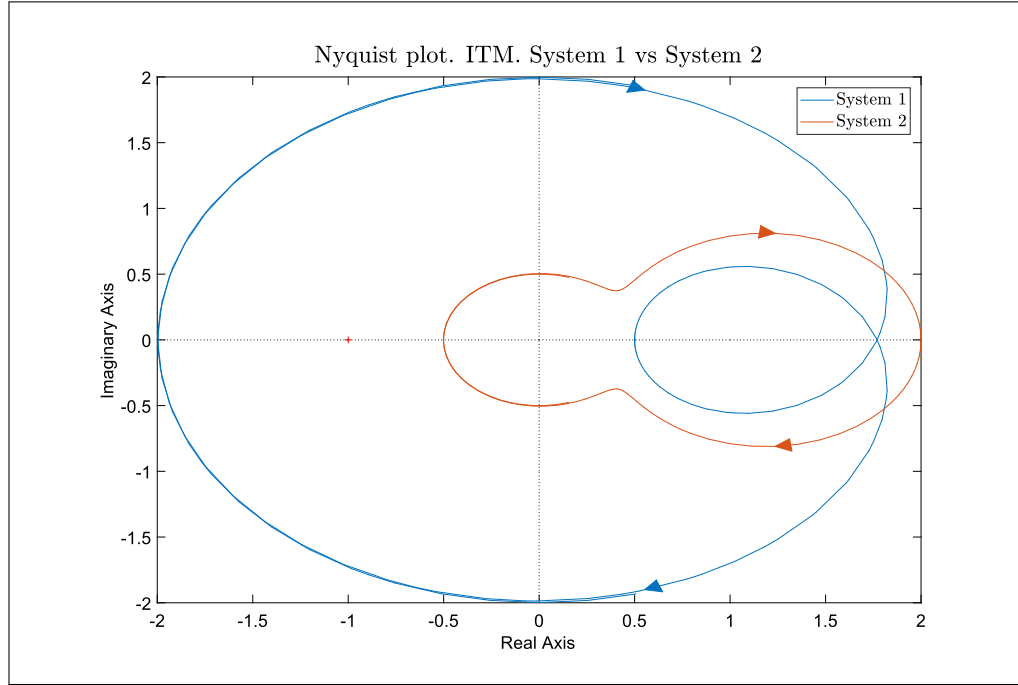


FIGURE 2.4. Comparison between stable and unstable cases of ITM algorithm. Nyquist diagram

TABLE 2.1. System Parameters

Parameter	System 1	System 2
R_s	1Ω	2Ω
R_{HUT}	2Ω	1Ω
L_s	2mH	1mH
L_{HUT}	1mH	2mH
t_a	$100 \mu s$	$100 \mu s$
t_b	$1 \mu s$	$1 \mu s$
Z_{ab}	$0,01\Omega + 10\mu H$	$0,01\Omega + 10\mu H$

Ways of finding more precise conditions for stability have been proposed in [Zhang \(2016\)](#) and [Brandl \(2017\)](#). However, both of these approaches have potential flaws.

TABLE 2.2. Stability Margins. ITM

	Gain margin	Phase margin
System 1. ITM	0.5 dB	-148.8 °
System 2. ITM	1.9 dB	137.4 °

In [Zhang \(2016\)](#) the author attempts to use the Routh-Hurwitz stability criteria to find conditions for stability based on the values of the impedances and the delay. To do this, it approximates the delay with a first order Pade approximation. This allows the author to find the poles of the system and then determine when it will be stable. However, the approximation of the delay introduces serious distortions that significantly affect the stability of the system as seen in figure 2.5. We can see that the behaviour of the phase is heavily distorted making the system appear more stable than it actually is.

Using higher order approximations makes for a more accurate description of the system but results in ever more complicated conditions for stability as the order of the characteristic equation of the system increases, reducing the practical value of the approach.

In [Brandl \(2017\)](#), the author plots the Nyquist contours of the systems while varying the relation between Z_s and Z_{HUT} to form a 3D image. This approach is helpful as it allows us to visualise the impact the change in ratio of these two variables has in the stability of the system, as we know that while the contour does not enclose the -1,0 point, the system will be stable. However, as we have seen previously, the ratio between Z_s and Z_{HUT} is usually not constant throughout a given frequency range, meaning that there isn't a single Nyquist contour associated to one value of this ratio. Thus, the results presented can be misleading.

As such, the best and most general way to address the stability of an IA is to rely on the standard tool to assess stability: Nyquist diagrams and gain and phase margins. This is the approach we will use throughout our analysis.

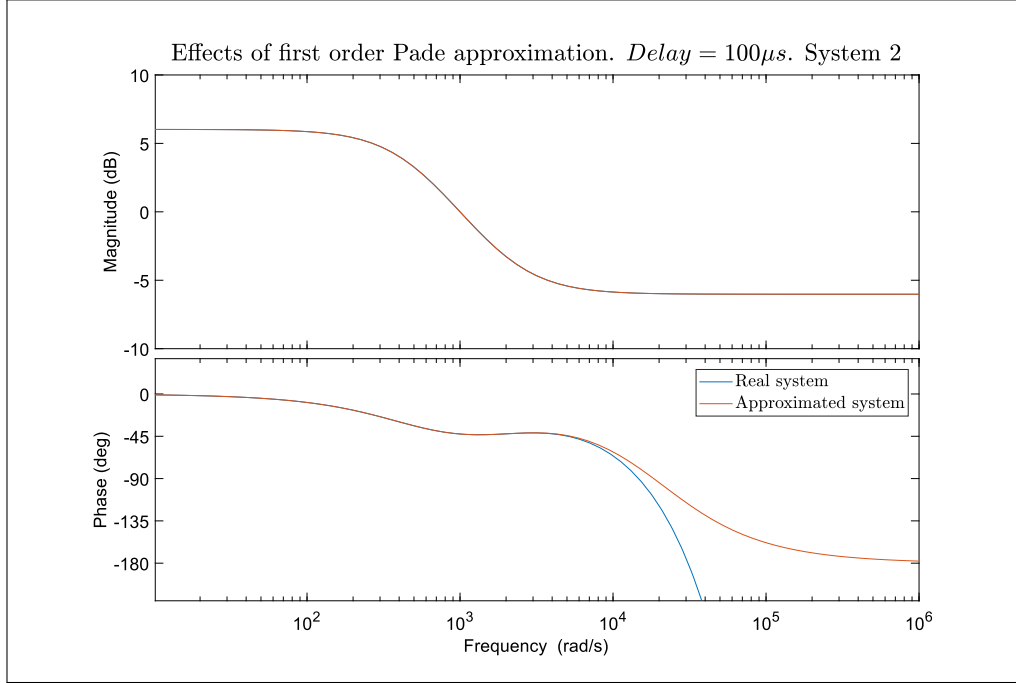


FIGURE 2.5. Difference between real system and system with delay approximation.

2.2.2. Accuracy considerations

The second most important objective of a PHIL IA is to provide accurate results. But what exactly do we mean by accurate? In this context, accuracy will be considered in relation to the real physical system, where the IA is not present. This system will be called the *Naturally coupled system* or NCS, for short. Given how we have chosen to represent our electrical system, the NCS will be nothing more than the voltage divider between Z_s and Z_{HUT} . We will use this as reference to compare the behaviour of the ITM algorithm. In order to compare all IAs fairly we must use the same system for all accuracy comparisons. This will be System 2, described in table 2.1 as it is stable with all IAs.

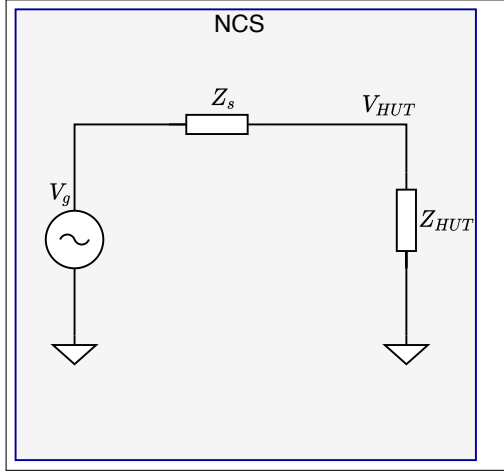


FIGURE 2.6. Naturally coupled system

Calculating transfer function from V_g to V_{HUT} is trivial, resulting in:

$$G_{NCS} = \frac{V_{HUT}}{V_g} = \frac{Z_{HUT}}{Z_s + Z_{HUT}} \quad (2.4)$$

As we wish for our PHIL simulation to mimic the behaviour of the NCS as closely as possible, we would like that the closed loop transfer function of the IA resembles the one of the NCS. The closed loop transfer function for the ITM algorithm can be calculated using the diagram shown in figure 2.2.

$$G_{cl} = \frac{G_a e^{-st_a} \frac{Z_{HUT}}{Z_{HUT} + Z_{ab}}}{1 + G_a e^{-st_a} \cdot G_b e^{-st_b} \cdot \frac{Z_s}{Z_{HUT} + Z_{ab}}} = \frac{G_a e^{-st_a} Z_{HUT}}{Z_{HUT} + Z_{ab} + Z_s \cdot G_a e^{-st_a} G_b e^{-st_b}} \quad (2.5)$$

Other than the effect of Z_{ab} , which ideally should be very small, we can see that if distortion introduced by the sensors and amplifier is low and delays are short, the results obtained will be the same as the NCS. We can also see that all the sources of inaccuracies come from hardware limitations and not from the interface algorithm itself. Because of this, the ITM is one of the most accurate of the interface algorithms.

The sensor and amplifier distortion functions, while at first might look like a mayor issue, generally have very limited impact in a PHIL simulation. Proper selection of components and good experiment design should make it so that these functions are generally very close to unity and thus, have little impact. Even in cases where this is not possible, such as in high power PHIL simulations where the amplifier may have low bandwidth and high input lag, these effects can be mitigated by active compensation as shown in (Ainsworth et al., 2016), (Marks et al., 2018a). For our platform however, as we will see in Chapter 5, we have chosen a linear amplifier with an extremely low output impedance, high bandwidth and low input lag, meaning that amplifier distortion should not be a problem.

This leaves delays as the most important source of inaccuracies within a PHIL simulation. In particular, the delay t_a , that represents the sum of the computation delay plus the amplifier input delay, will be the main source of error as it will likely be orders of magnitude larger than the sensor delay.

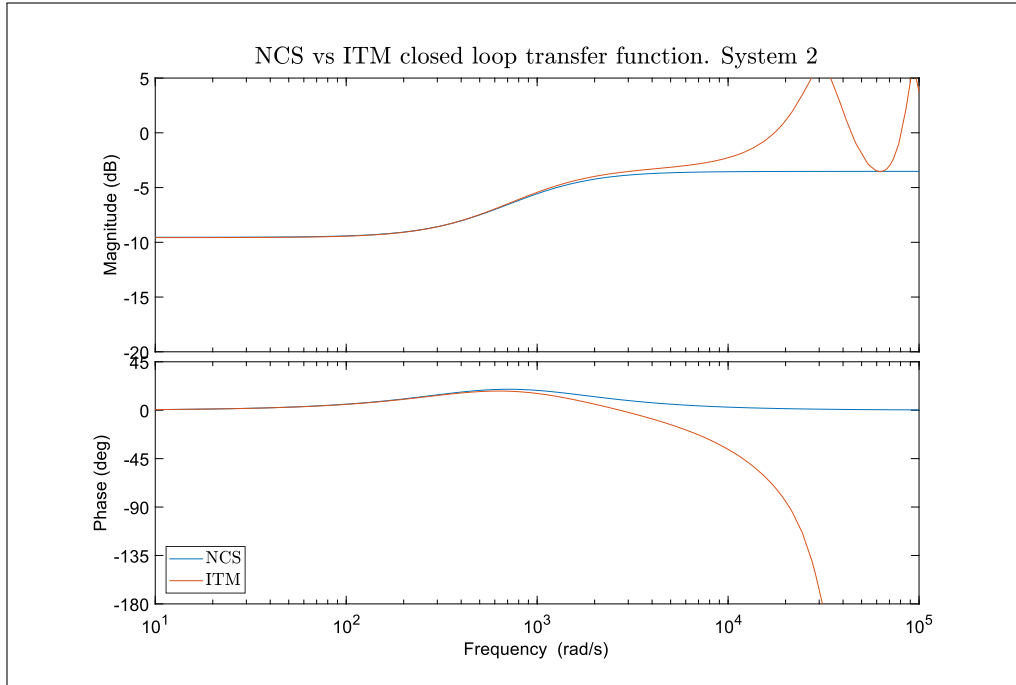


FIGURE 2.7. NCS vs ITM

The main effect of the delay is to make the phase of the system increase with frequency. This means that any behaviour of the system at frequencies higher than that of the delay, will not be represented accurately. These effects are not limited to the phase though, as the delay can cause oscillations in the magnitude of the closed loop transfer function as frequency increases past the period of the delay, as seen in figure 2.7. To avoid any adverse effect, it is crucial that all the relevant dynamics of the simulation are within the bandwidth set by the delay's period. Most the time, these spurious high frequency dynamics will be filtered out naturally due to the limited bandwidth of the amplifier and/or sensor. Regardless, it is still good practice to place a low pass filter somewhere within the control loop with a cut off frequency just short of the delay's. The addition of a low pass filter in the feedback path of the system can also increase stability, as seen in the next section.

The errors with respect to the NCS can be more clearly seen in figure 2.8. Here we show the difference between the two transfer functions. We also plot a 0.5dB range, corresponding to $\pm 5.92\%$ of deviation with respect to the NCS.

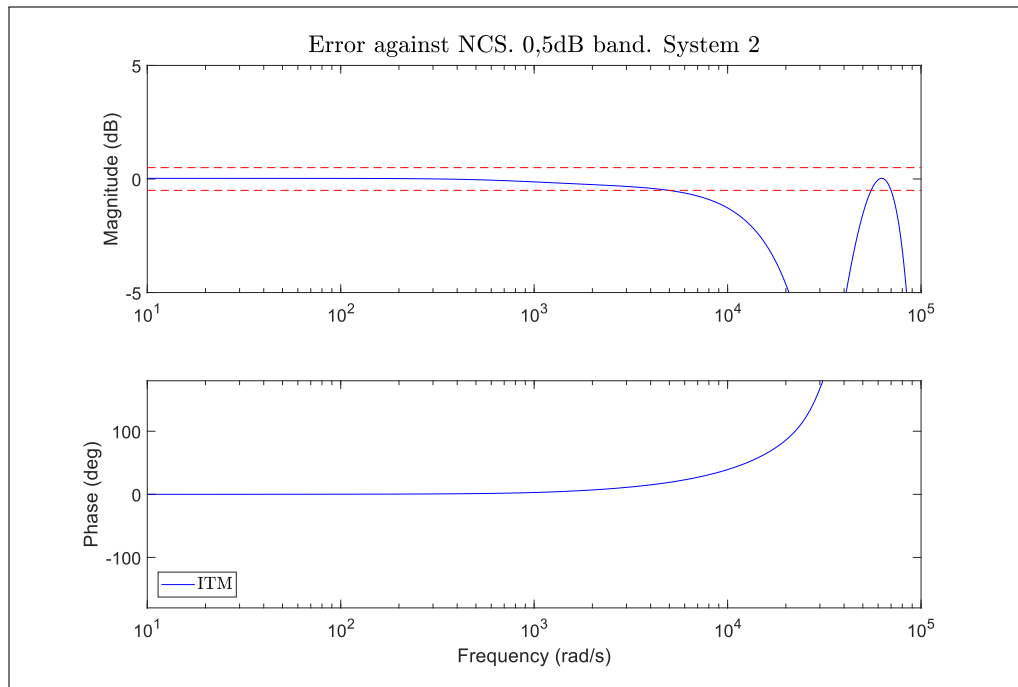


FIGURE 2.8. ITM. Deviations with respect to NCS

2.3. Ideal transformer method with low pass filter

This interface algorithm is a straight forward approach to the stability problems of the ITM. It deals with them by adding a low pass filter to the current feedback path in the standard ITM. The stabilising effect from this filter comes from reducing the magnitude of the open loop transfer function at high frequencies, so that by the time the delay pushes the phase of the system below -180° , it is already below 0dB. The open loop transfer function can be derived from the system's block diagram, just like the prior case. The control diagram can be seen in figure 2.9. Note the addition of G_{LPF} in the feedback path.

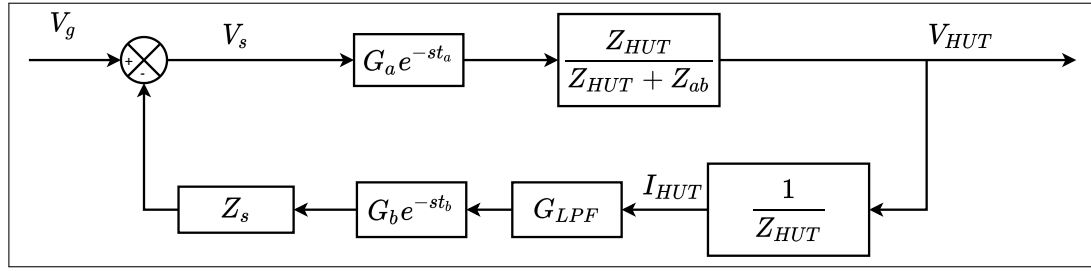


FIGURE 2.9. ITM method with lowpass filter. Block diagram

2.3.1. Stability considerations

The effects the filter has in the stability can be seen in the figures 2.10 and 2.11 where we compare the Bode and Nyquist plots of the regular ITM and the ITM with a low pass filter. We use system 1 from the previous section as it was unstable for the regular ITM, however we have managed to stabilise it with a second order low pass filter with a cut off frequency of 1200 Hz. We can appreciate how in the Nyquist contour the higher frequencies are attenuated, preventing the contour from encircling the $(-1,0)$ point. The effects of the filter are even more evident in the stability margins of the system, seen in table 2.3.

In practice, this method is very straight forward to implement as it only requires a rough estimation of the plant's impedance and knowledge of the system's delay in order to design a filter that can stabilise the system.

Nonetheless, this method is not without drawbacks, as the lowpass filter will change the behaviour of the system at high frequency which will results in loss of simulation accuracy. In particular, it can be seen as an increase in the mismatch between the simulation and the HUT active and reactive power as the effect will compound with the one produced by the PHIL simulation's delay ([Ainsworth et al., 2016](#)).

TABLE 2.3. Stability Margins. ITM with low pass filter - System 1

	Gain margin	Phase margin
ITM	0.5 dB	-148.8 °
ITM with lowpass filter	1.2 dB	26.4 °

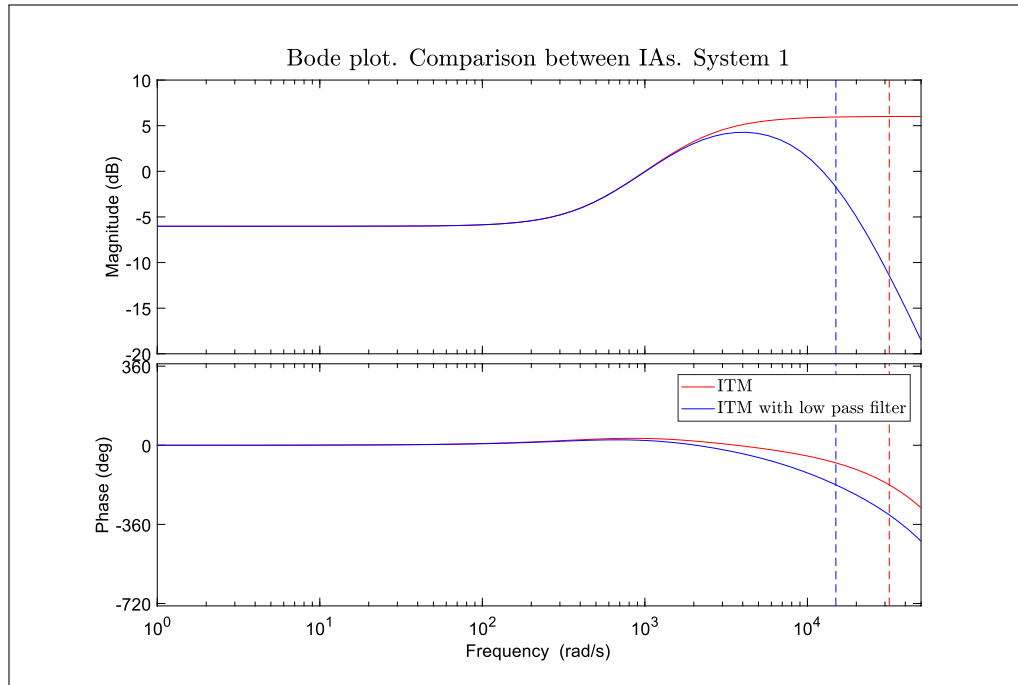


FIGURE 2.10. Comparison between ITM and ITM with lowpass filter IAs

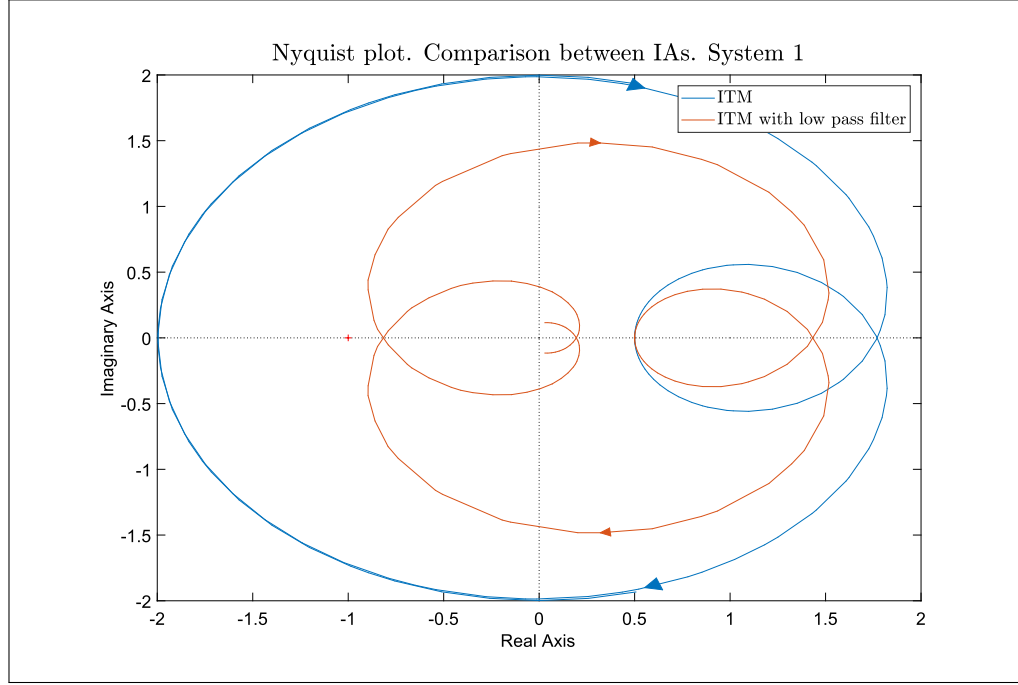


FIGURE 2.11. Comparison between ITM and ITM with lowpass filter IAs.

2.3.2. Accuracy considerations

To evaluate the accuracy of this method we will again use the closed loop transfer function of the system. This function can be derived from diagram 2.9 and as expected, it is similar to that of the normal ITM but with the addition of the low pass filter in the denominator.

$$G_{cl} = \frac{G_a e^{st_a} Z_{HUT}}{Z_{HUT} + Z_{ab} + Z_s \cdot G_a e^{st_a} G_b e^{st_b} \cdot G_{LPF}} \quad (2.6)$$

The impact of the lowpass filter on the systems accuracy can be seen in figures 2.12 and 2.13. Here, a 1200 Hz lowpass filter was applied to the system in case 1 and its closed loop transfer functions is compared with the regular ITM and the NCS system. We see two main effects. First, the bandwidth of the system has been reduced by the low pass filter. Previously, the behaviour of the ITM was similar to that of the NCS until the influence of

the delay came into play. Now, the magnitude of the system starts differing from the NCS much sooner as a consequence of the low pass filter. This is clearly visible in figure 2.13 where the ITM with lowpass filter leaves the 0.5 dB error band almost a full decade sooner than the regular ITM.

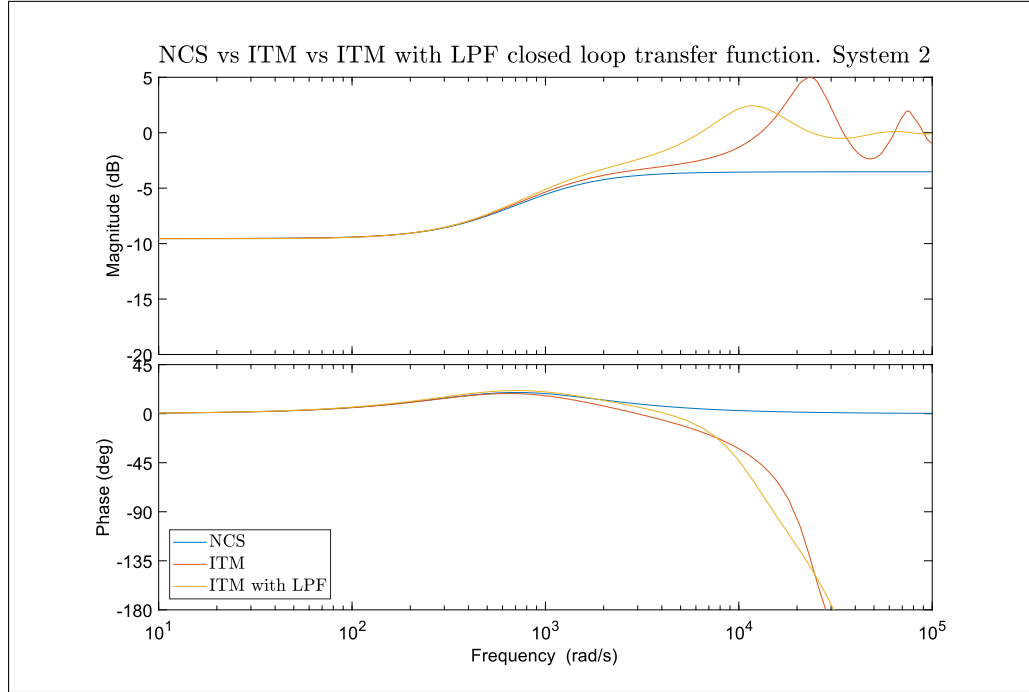


FIGURE 2.12. Closed loop response of system with low pass filter compared to ITM and NCS systems

A second effect is the attenuation of the oscillations in the magnitude at higher frequencies, present in the regular ITM. A difference in the magnitude of the systems is still observed between the ITM with a low pass filter and the NCS. For the NCS, the magnitude will be given by the inductance ratio, producing the corresponding attenuation or amplification of the high frequencies. By contrast, the magnitude of these higher frequencies for the ITM with a lowpass filter will be 0dB, meaning that they will not be attenuated nor amplified. This may sound counter intuitive at first, but it must be remembered that the lowpass filter is in the feedback path and as such it will prevent high frequency signals

from propagating back and affect the system's stability but it will not prevent these signals from reaching the output if they are present in the reference.

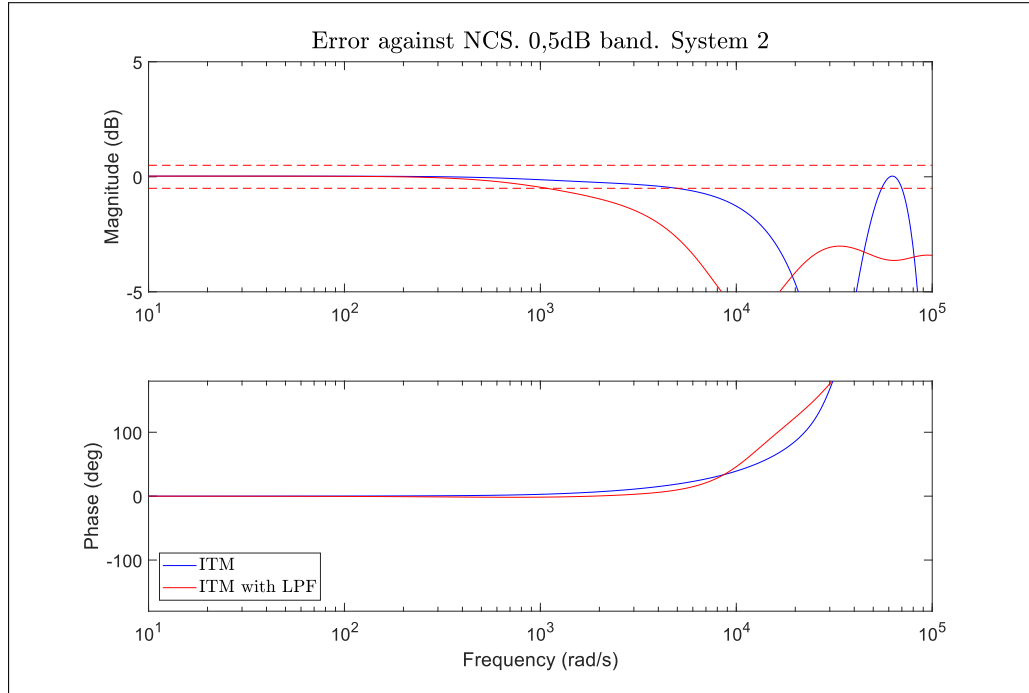


FIGURE 2.13. ITM with low pass filter. Deviations with respect to NCS

2.4. Partial circuit duplication (PCD)

The partial circuit duplication algorithm is based on V-type interface used by relaxation based circuit solvers shown in [Newton y Sangiovanni-Vincentelli \(1984\)](#). In this algorithm, the interface consist on two voltage sources, each controlled by the voltage of the opposite side. A linking component is Z_{ab} is also added. The structure of the PCD algorithm can be seen in figure 2.14. The PCD algorithm shows outstanding characteristics regarding stability, however, it has a lacklustre performance in terms of accuracy and ease of implementation.

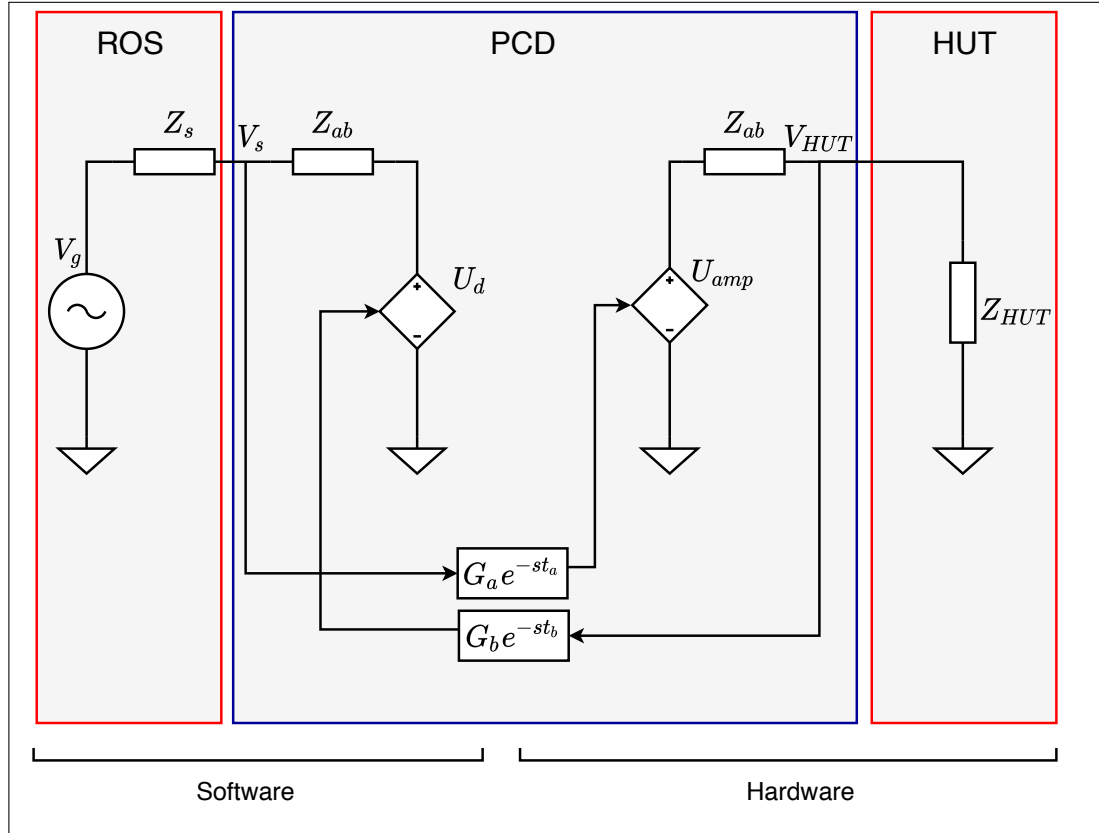


FIGURE 2.14. PCD algorithm

2.4.1. Stability considerations

As with previous interface algorithms, to further analyse the stability of the PCD we will first obtain its control block diagram and corresponding open loop transfer function.

Example 2.2. Let us start with V_s . This voltage is taken by the voltage amplifier U_{amp} and it is applied to the hardware side. There, V_{HUT} will be the result of the voltage divider between Z_{HUT} and Z_{ab} .

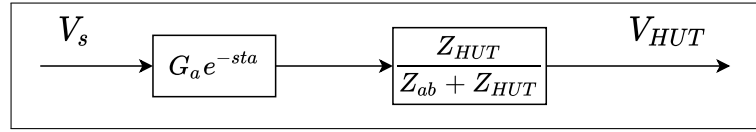


FIGURE 2.15. PCD block diagram: forward path

This voltage is then sensed and applied back to the simulation through U_d . Now, to form V_s we see that it will be the sum of the contributions made by V_g and U_d .

$$V_s = V_g \cdot \frac{Z_{ab}}{Z_s + Z_{ab}} + U_d \cdot \frac{Z_s}{Z_s + Z_{ab}} \quad (2.7)$$

Using equation 2.7 we can complete the block diagram for the PCD algorithm.

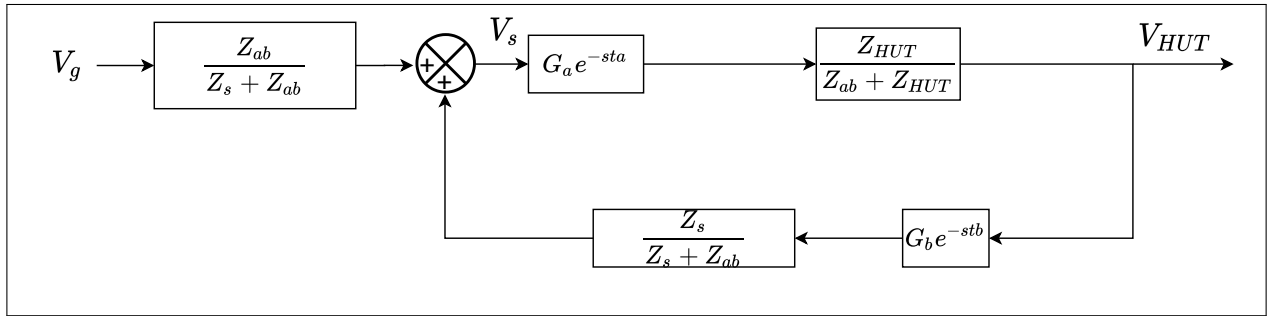


FIGURE 2.16. PCD block diagram: complete

The diagram obtained in example 2.2 allows us to easily obtain the open loop transfer function. Unlike the other two methods studied thus far, the diagram for the PCD algorithm

has a feedforward part. This part can be neglected when calculating the open loop transfer function as it is not part of the feedback loop. With this in mind, we can calculate the open loop transfer by simply multiplying the blocks in the loop. This yields the following function. We must also keep in mind that the PCD has a positive feedback loop rather than a negative one. Thus, to analyse its stability and compare it with the previous algorithms we will analyse $-G_{OL}$.

$$-G_{OL} = -G_a e^{-st_a} \cdot G_b e^{-st_b} \frac{Z_s \cdot Z_{HUT}}{(Z_{ab} + Z_s)(Z_{ab} + Z_{HUT})} \quad (2.8)$$

By inspecting equation 2.8, we can notice something remarkable; unless G_a or G_b make it so, it is impossible for the magnitude of the open loop transfer function to have a value above 1. This means that unless the distortions of the amplifier or sensors amplify the signals in some way it is impossible for the PCD algorithm to be unstable. This is because the rightmost part of the transfer function is always less than 1. This makes the PCD the most stable of the interface algorithms.

$$\left\| \frac{Z_s \cdot Z_{HUT}}{(Z_{ab} + Z_s)(Z_{ab} + Z_{HUT})} \right\| < 1 \quad (2.9)$$

While the transfer function's gain can theoretically never be more than one, the gain margin can be still be close to 0dB if Z_{ab} is very small, putting the system dangerously close to being unstable, as seen in figure 2.18. The key to the PCD's stability is the linking component, as increasing the value of Z_{ab} , will always reduce the gain of the open loop transfer function. The larger the impedance of this component, the more stable the system will be.

We also see, as shown in figure 2.17, that the resistive and inductive parts of the linking impedance have different effects on the system. A large resistive component (blue curve) will reduce the gain of the open loop transfer function by a lot at DC and low frequencies but will have a limited effect at the high end of the spectrum if both Z_s and Z_{hut} have an

inductive part. This, as the impedance of the inductive part will increase with frequency while the resistive part stays the same. This can be seen in the nyquist diagram as the gain approaches 1 at high frequencies while forming a circle due to the influence of the delay.

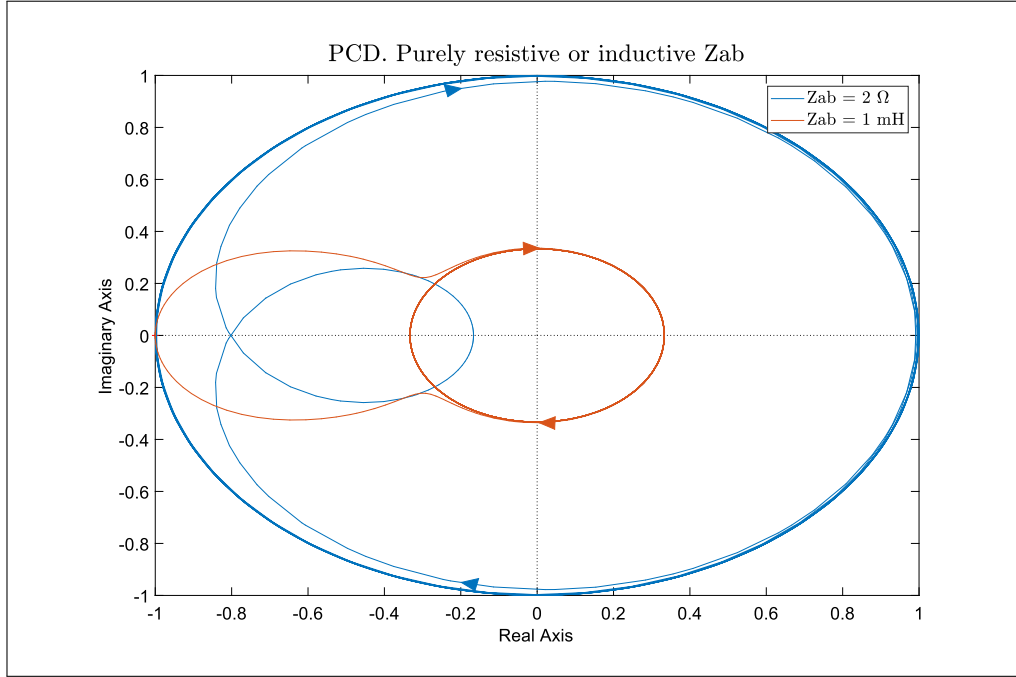


FIGURE 2.17. PCD. Comparison between purely resistive and purely inductive Z_{ab}

The inductive part (orange curve) will have the opposite behaviour, having negligible effects at low frequencies causing the DC gain to be 1. As frequency increases, the gain of the open loop transfer function will be given by the following ratio: $\frac{L_s \cdot L_{HUT}}{(L_{ab} + L_s)(L_{ab} + L_{HUT})}$.

As such, to use the PCD algorithm effectively, one must choose Z_{ab} to have a resistive part big enough in relation to those of Z_s and Z_{HUT} to make sure the DC gain has a relatively safe gain margin, while also choosing a topology similar to that of Z_s and Z_{HUT} to ensure that as frequency grows, the impedance of Z_{ab} follows the behaviour of the other two, and thus we have a gain smaller than 1 throughout the frequency spectrum without sacrificing accuracy.

Figure 2.18 shows the the nyquist curves of the PCD algorithm for different values of Z_{ab} while the values for the stability margins for each case can be seen in table 2.4. The values of Z_{ab} are a multiple of the base Z_{ab} used for the rest of the algorithms. We see how low values of Z_{ab} make for very slim stability margins. As the value of this component increases, so does the stability of the system. However, as seen in the next section, large values for Z_{ab} will result in a loss of accuracy, thus a trade off has to be made between large stability margins and high accuracy. Figure 2.19 compares an instance of the PCD where Z_{ab} was chosen to give reasonable stability margins, while sacrificing as little accuracy in the process, to the previous IAs we have reviewed.

TABLE 2.4. Stability Margins. PCD - System 1

	Gain margin	Phase margin
$Z_{ab} \cdot 1$	0.13 dB	INF
$Z_{ab} \cdot 3$	0.386 dB	INF
$Z_{ab} \cdot 10$	1.25 dB	INF
$Z_{ab} \cdot 30$	3.22 dB	INF
$Z_{ab} \cdot 100$	8.55 dB	INF
$Z_{ab} \cdot 300$	18.5 dB	INF

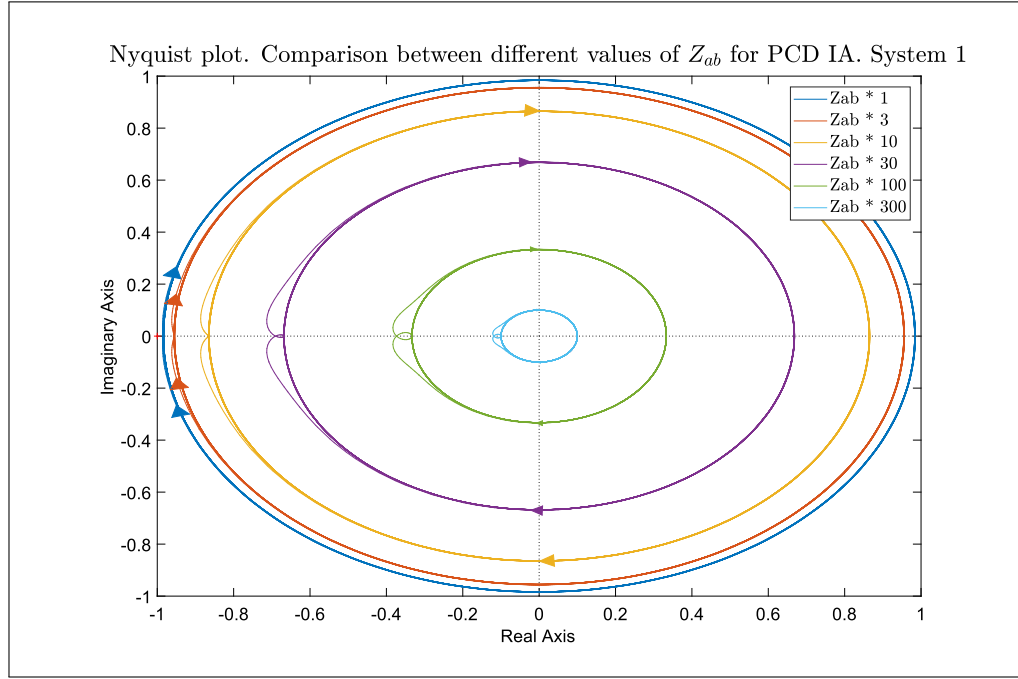


FIGURE 2.18. Nyquist contours of PCD algorithm. Multiple values for Z_{ab}

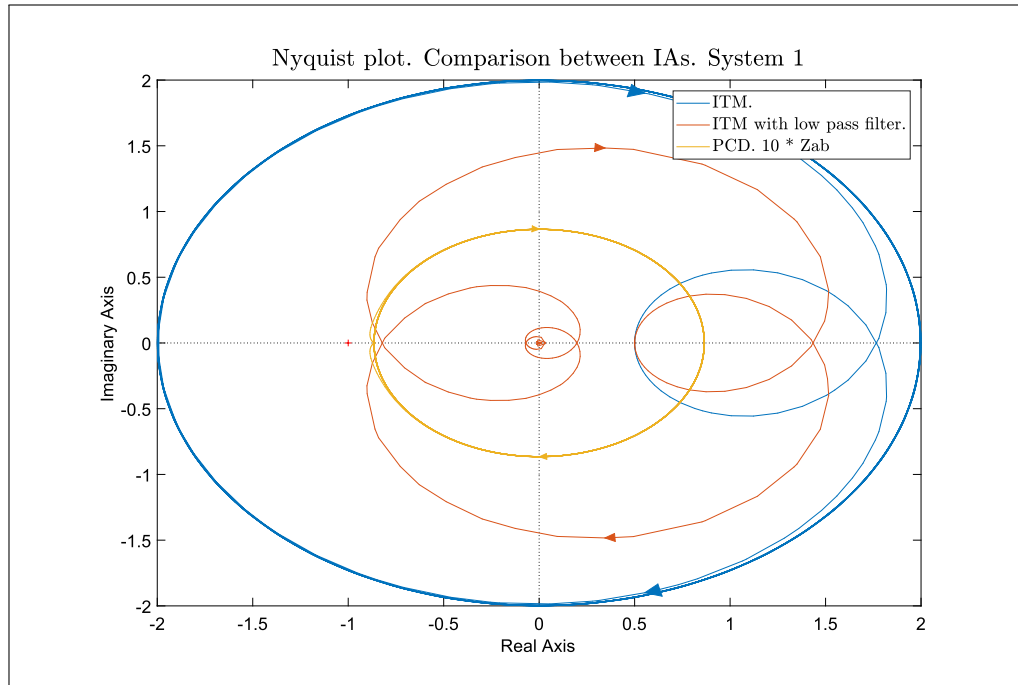


FIGURE 2.19. PCD vs ITM vs ITM with lowpass filter

2.4.2. Accuracy considerations

The main drawback of the PCD algorithm is its poor accuracy (Brandl, 2017). The reason is that the stability of the IA depends on the magnitude of the linking impedance, the larger the magnitude, the more stable the system. But the value Z_{ab} takes, greatly affects the voltage tracking of the IA. Like the previous methods, we will use the closed loop transfer function of the IA to study its accuracy.

Example 2.3. The diagram shown in figure 2.15 can be simplified by defining transfer functions F , G and H . Each corresponding to one part of the loop, as shown in figure 2.20.

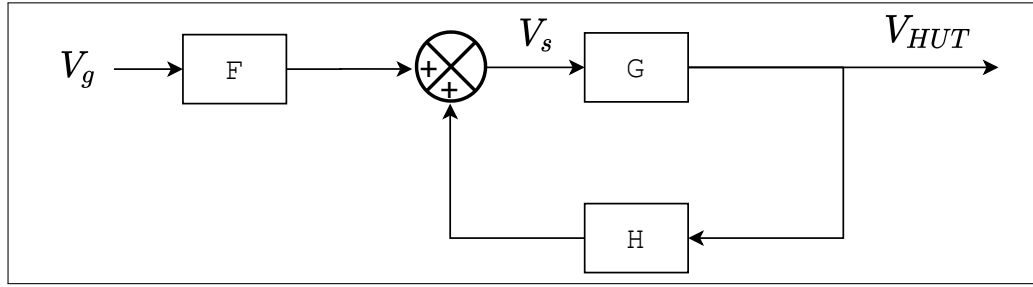


FIGURE 2.20. Simplified control loop

This allows for the calculation of the closed loop transfer function using equation 2.13. Note that the sign in the denominator of the transfer function is negative due to the positive sign in the feedback loop.

$$F = \frac{Z_{ab}}{Z_{ab} + Z_s} \quad (2.10)$$

$$G = \frac{Z_{HUT}}{Z_{ab} + Z_{HUT}} G_a e^{-st_a} \quad (2.11)$$

$$H = \frac{Z_s}{Z_{ab} + Z_s} G_b e^{-st_b} \quad (2.12)$$

$$G_{CL} = \frac{F \cdot G}{1 - G \cdot H} \quad (2.13)$$

Using this, the complete closed loop transfer function of the PCD algorithm is shown in equation 2.14.

$$G_{CL} = \frac{Z_{ab} \cdot Z_{HUT} \cdot G_a \cdot e^{-s(t_a)}}{(Z_{ab} + Z_s)(Z_{ab} + Z_{HUT}) - Z_s \cdot Z_{HUT} \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)}} \quad (2.14)$$

Unlike the previous IAs, it is not clear from equation 2.14 that the PCD should produce accurate results under ideal conditions. Nonetheless, this is the case. To show this, we will assume no delays and no distortion.

Example 2.4. Let $G_a = G_b = 1$ and $t_a = t_b = 0$. With these ideal conditions equation 2.14 can be further simplified.

$$G_{CL} = \frac{Z_{ab} \cdot Z_{HUT}}{(Z_{ab} + Z_s)(Z_{ab} + Z_{HUT}) - Z_s \cdot Z_{HUT}} \quad (2.15)$$

$$G_{CL} = \frac{Z_{ab} \cdot Z_{HUT}}{(Z_{ab}^2 + Z_{ab} \cdot Z_s + Z_{ab} \cdot Z_{HUT} + Z_s \cdot Z_{HUT} - Z_s \cdot Z_{HUT})} \quad (2.16)$$

$$G_{CL} = \frac{Z_{HUT}}{Z_{ab} + Z_s + Z_{HUT}} \quad (2.17)$$

From equation 2.17 it becomes clear that if $Z_{ab} = 0$, then the transfer function will equal that of the NCS.

Example 2.4 illustrates the main disadvantage of the PCD IA. Even in the ideal scenario, there is an inherent trade off between stability and accuracy. The larger Z_{ab} is, the more stable the system but also the more inaccurate. The problem is ever more pronounced when the effects of the delay are considered. Figure 2.21 shows the closed loop functions of the PCD algorithm for different values of Z_{ab} . While it remains true that in general, increasing Z_{ab} makes the system less accurate, very low values of Z_{ab} can have problems of accuracy as well. We see how the case when $Z_{ab} * 1$ is good at low frequencies but the accuracy worsens considerably at the high end of the spectrum. It is instead $Z_{ab} * 10$

the case that shows the best performance overall across the frequency range, as seen in figures 2.21 and 2.22.

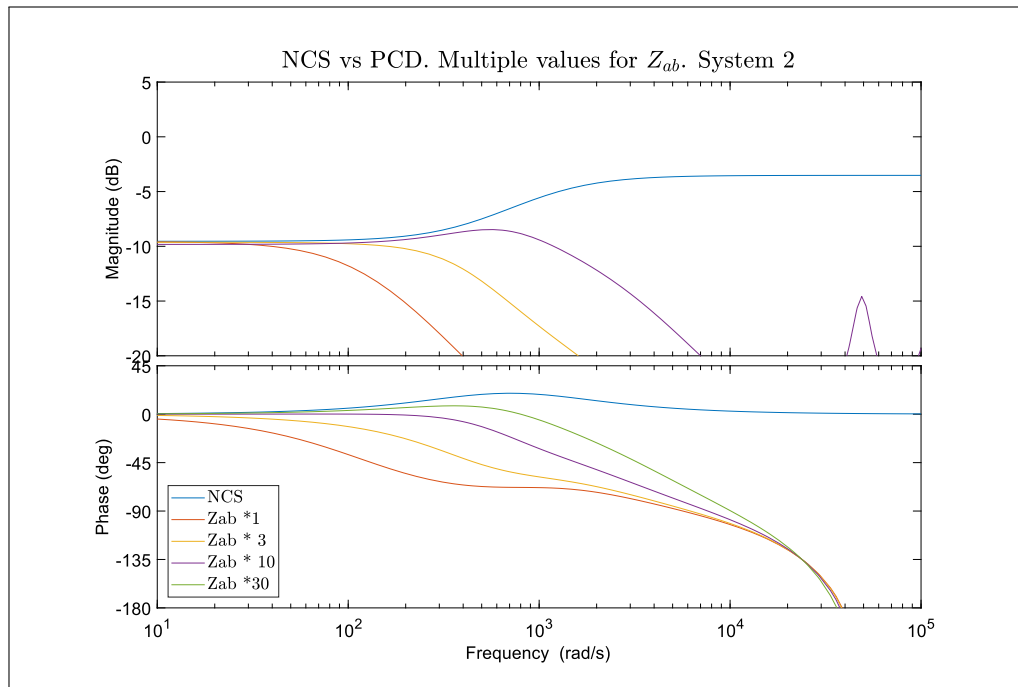


FIGURE 2.21. PCD accuracy. Multiple values

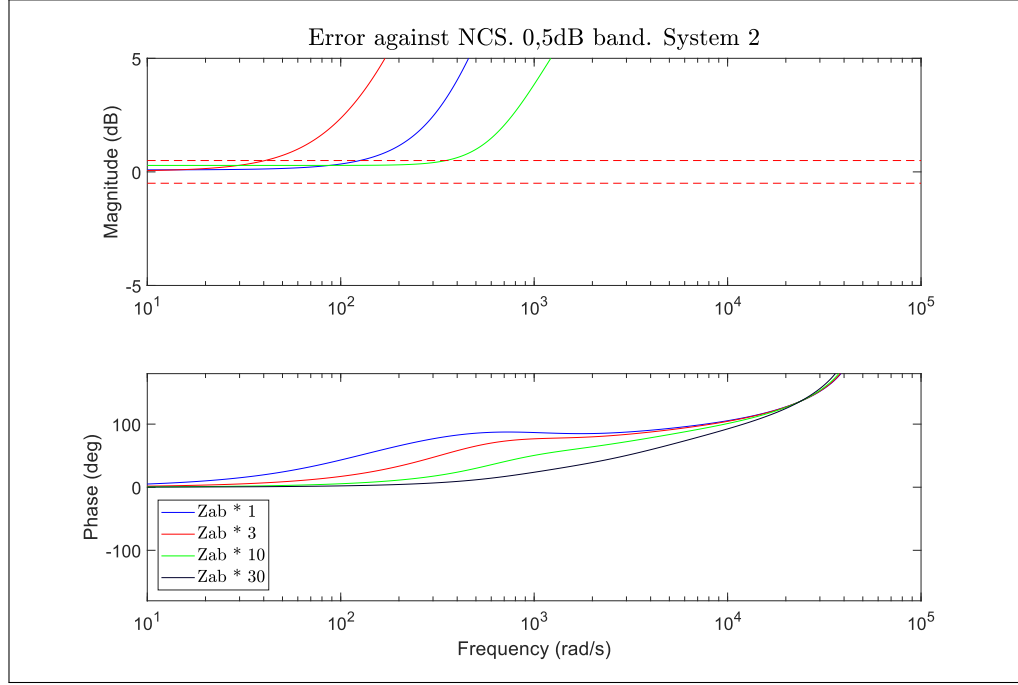


FIGURE 2.22. PCD. Multiple values of Z_{ab} . Deviations with respect to NCS

The performance compared to the previous interface algorithms is also lacklustre. While any PHIL experiment using the PCD is almost guaranteed to be stable if Z_{ab} is sufficiently large to account for any problems the distortion functions might cause, this will come at the cost the experiment's accuracy. Figures 2.23 and 2.24 compare the performance of the PCD with in our best case scenario, with the IAs previously studied. We see that even the best case of the PCD performs worse than the ITM or even the ITM with lowpass filter, being accurate in a narrower band of band of frequencies. We also see that even in this band of frequencies there are small magnitude inaccuracies, product of the non-zero values of Z_{ab} . The performance of the phase of the system is even worse, presenting significant error much earlier than the other two algorithms. As such the PCD, while very stable, will often fail to produce useful results in a PHIL simulation.

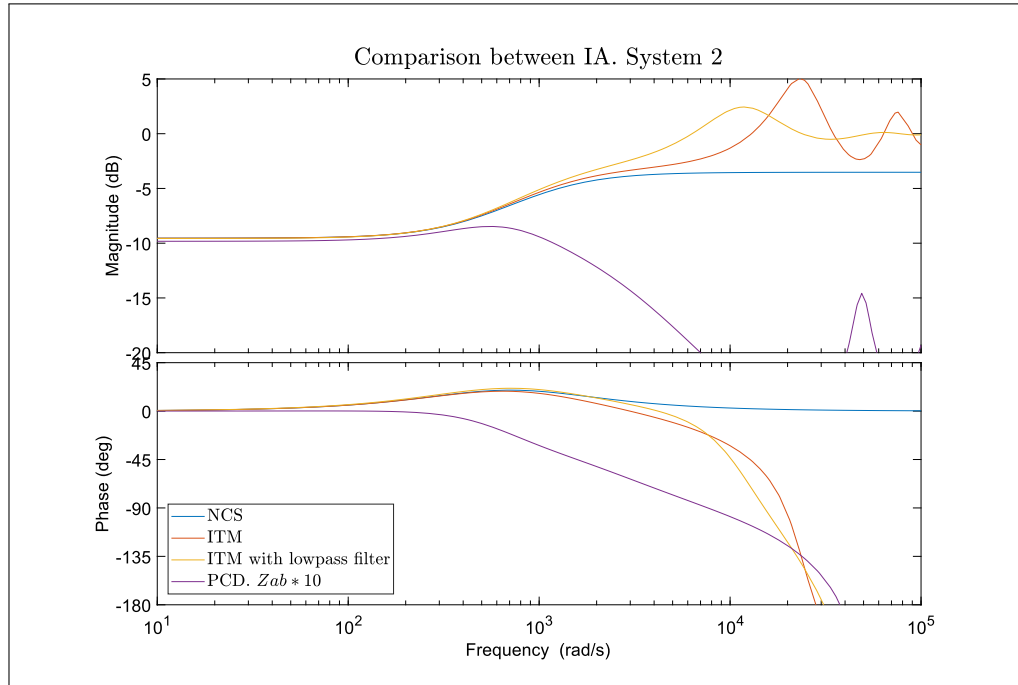


FIGURE 2.23. PCD accuracy vs other interface algorithms

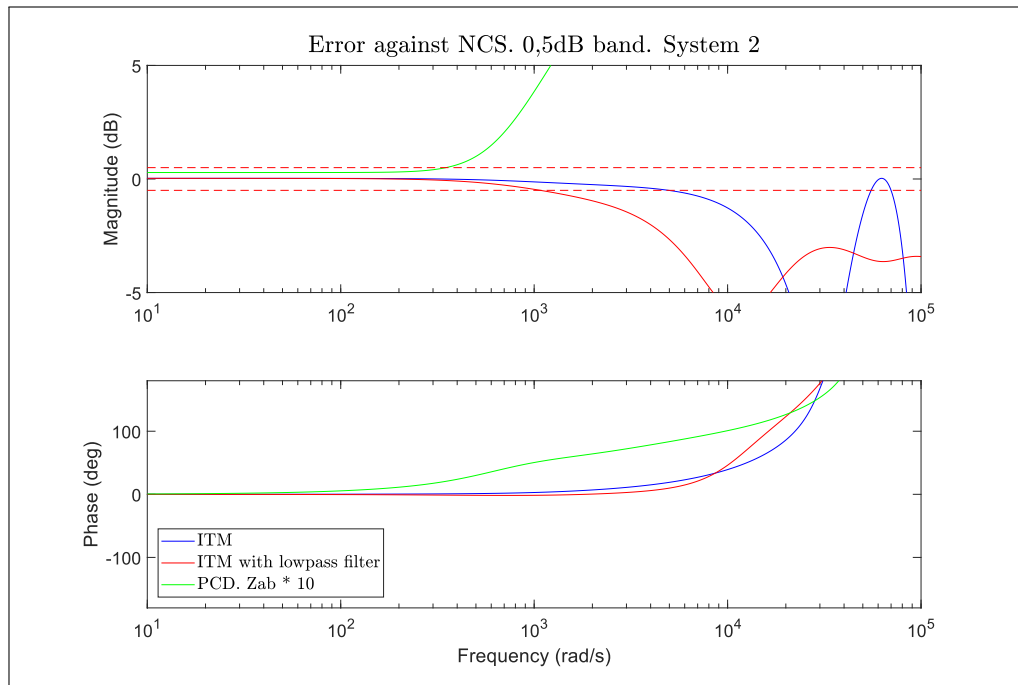


FIGURE 2.24. PCD. Deviations with respect to NCS

2.5. Damping Impedance method (DIM)

So far, none of the methods studied have provided the combination of stability and accuracy needed for a reliable PHIL platform as they presented us with a trade-off between the two. There is however, a third interface algorithm derived from the hybrid I-V type interface, presented in [Newton y Sangiovanni-Vincentelli \(1984\)](#). This IA, called the damping impedance method (DIM), can be understood as a combination of the ITM and PCD via the addition of a damping impedance Z_{DIM} , as seen in figure 2.25. By inspecting the algorithm we notice that the value of this impedance will greatly impact the behaviour of the algorithm in terms of stability and accuracy. If the damping impedance value is zero, then the algorithm will behave like the PCD as the voltage U_d will be equal to V_s , assuming Z_{ab} is very small. If Z_{DIM} is infinity, we can effectively eliminate that branch of the model and we are left with the ITM algorithm. Thus, we can conclude that the properties of the DIM algorithm will vary between these two extremes.

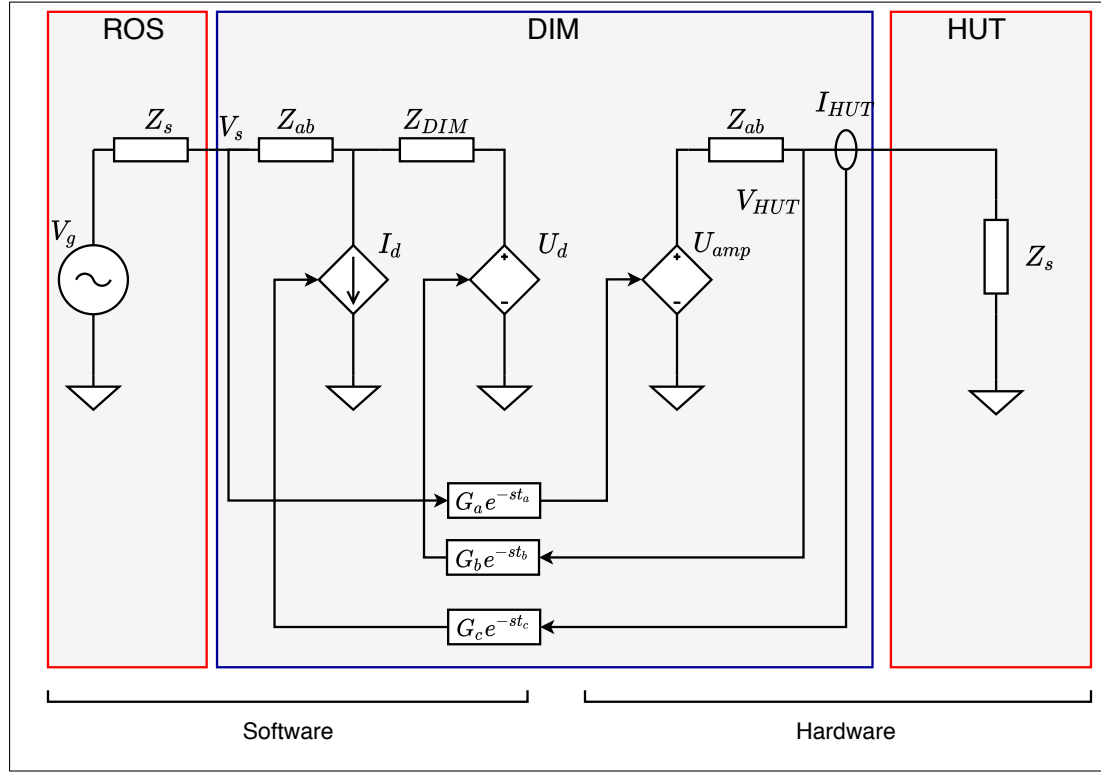


FIGURE 2.25. DIM algorithm

In fact, the DIM algorithm provides us with a solution to do away with the trade-off between accuracy and stability. This gain is not without cost though, as it requires significant effort to implement.

2.5.1. Stability considerations

To analyse in detail the stability of the DIM IA and the effects Z_{DIM} has on it, we will once again study the open loop transfer function of the system. To do this, we must first obtain the control block diagram.

Example 2.5. Let us start with the value of V_s in figure 2.25. This voltage is sampled and used as input for the amplifier. As such it becomes U_{amp} by passing through $G_a e^{-st_a}$.

Using U_{amp} , I_{HUT} can be calculated simply using the Ohm law. V_{HUT} can now be obtained by multiplying the current by Z_{HUT} .

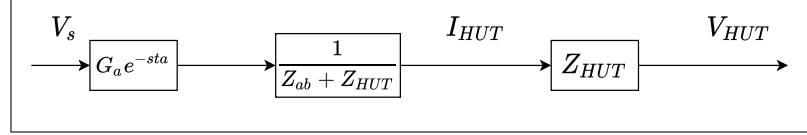


FIGURE 2.26. DIM forward path

With V_{HUT} and I_{HUT} , we can now obtain U_d and I_d in the left side of the circuit. This is done by multiplying them by the sensor's transfer functions: $G_b e^{-stb}$ and $G_c e^{-stc}$. Finally V_s can be calculated by superposition, separating the left side of the DIM algorithm in three circuits and adding the contributions from V_g , I_d and U_d .

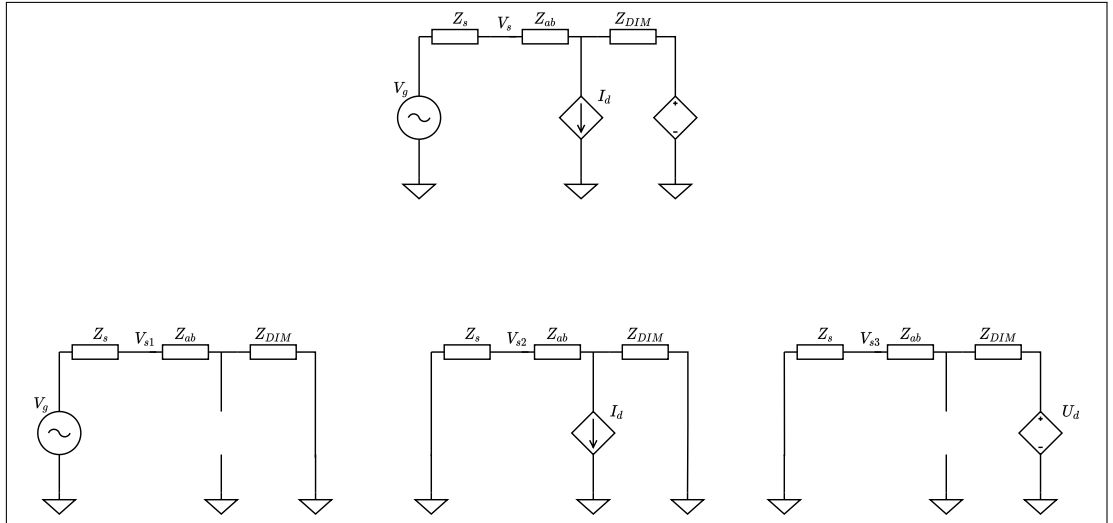


FIGURE 2.27. DIM left side solved by superposition

The contributions from V_g and U_d are easy to calculate as they are voltage dividers. The contribution from I_d , by contrast, is obtained by calculating the voltage at the current source and then subtracting the voltage drop at Z_{ab} .

$$V_{s1} = \frac{Z_{DIM} + Z_{ab}}{Z_s + Z_{ab} + Z_{DIM}} \cdot V_g \quad (2.18)$$

$$\begin{aligned} V_{s2} &= - \left(\frac{(Z_s + Z_{ab}) \cdot Z_{DIM}}{Z_s + Z_{ab} + Z_{DIM}} - \frac{Z_{ab} \cdot Z_{DIM}}{Z_s + Z_{ab} + Z_{DIM}} \right) \cdot I_d \\ &= - \frac{Z_s \cdot Z_{DIM}}{Z_s + Z_{ab} + Z_{DIM}} \cdot I_d \end{aligned} \quad (2.19)$$

$$V_{s3} = \frac{Z_s}{Z_s + Z_{ab} + Z_{DIM}} \cdot U_d \quad (2.20)$$

$$V_s = v_{s1} + v_{s2} + v_{s3} \quad (2.21)$$

The calculation of V_s allows us to draw the feedback paths of the control diagram, thus completing it.

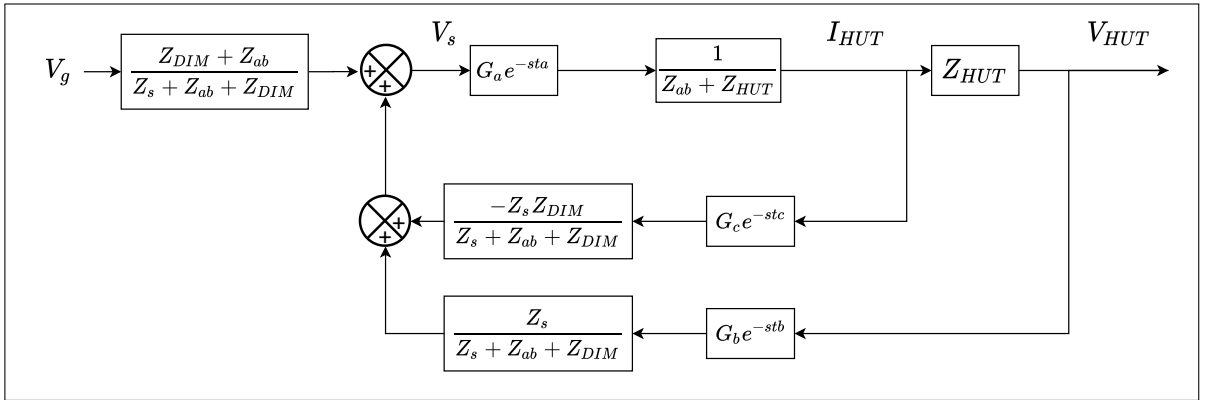


FIGURE 2.28. DIM control block diagram

From the block diagram shown in figure 2.28 we can easily calculate the open loop transfer function of the system to analyse the stability of the system.

Example 2.6. To calculate the open loop transfer function we must first make a few changes to the diagram obtained in example 2.5, as seen in figure 2.29. This change allows us to sum both feedback paths.

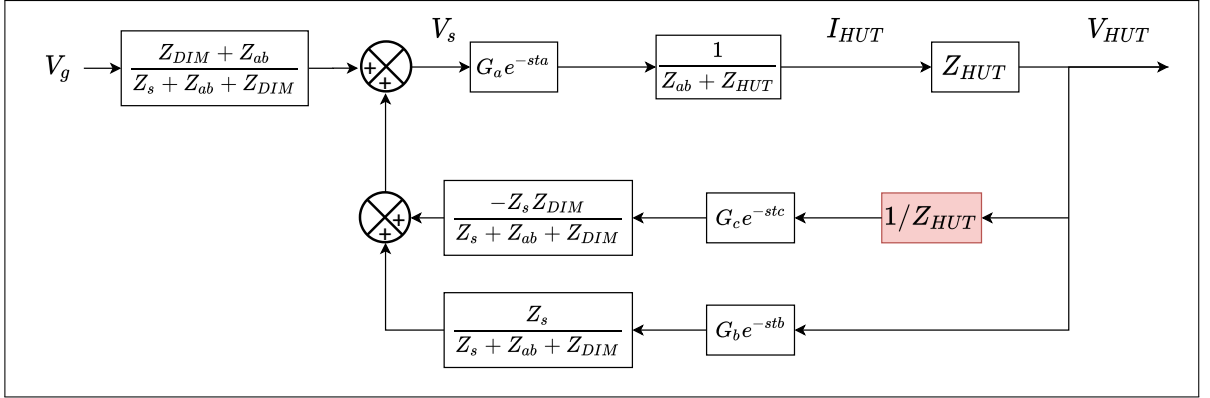


FIGURE 2.29. DIM control block diagram: modified

We can now apply the same method used to calculate the transfer function of the PCD, shown in example 2.3 to simplify the control block diagram, as seen in figure 2.20. Defining the transfer functions F, G and H , for each part of the loop we get:

$$F = \frac{Z_{DIM}}{Z_s + Z_{ab} + Z_{DIM}} \quad (2.22)$$

$$G = \frac{G_a e^{st_a} \cdot Z_{HUT}}{Z_{ab} + Z_{HUT}} \quad (2.23)$$

$$\begin{aligned} H &= \frac{G_b e^{st_b} \cdot Z_s}{Z_s + Z_{ab} + Z_{DIM}} + \frac{-G_c e^{st_c} \cdot Z_s \cdot Z_{DIM}}{Z_{HUT} \cdot (Z_s + Z_{ab} + Z_{DIM})} \\ &= \frac{G_e e^{st_e} \cdot Z_s \cdot (Z_{HUT} - Z_{DIM})}{Z_{HUT} \cdot (Z_s + Z_{ab} + Z_{DIM})} \end{aligned} \quad (2.24)$$

The assumption that the sensor delays and distortions are equal ($G_b e^{-st_b} = G_c e^{-st_c} = G_e^{-st_e}$) allows us to further simplify H . While this may not be necessarily true in practice, the distortion and delay introduced by the sensors tend to have a negligible effect when compared to the distortion and input delay of the amplifier. Also, Depending on the complexity of the simulation, the computational delay might be large as well, when compared to the sensor delay. Thus, the effects of this simplification are likely to be very minor.

To calculate the open loop transfer we must break the control loop just before the feedback point and multiply the function along it. Since feed-forward part of the system, represented by F , is not part of the feedback loop, we exclude it from the calculation as it has no effect in the stability of the system. With this, the open loop transfer function will be $G \cdot H$.

$$G_{ol} = G \cdot H = Z_s \cdot G_e e^{st_e} \cdot G_a e^{st_a} \frac{(Z_{HUT} - Z_{DIM})}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM})} \quad (2.25)$$

The open loop transfer function seen in 2.25 is particularly interesting as quick inspection will reveal that if $Z_{HUT} = Z_{DIM}$, the function will be zero, regardless of simulation delay. This is important as the open loop transfer function being zero guarantees the stability of the IA as errors caused during one simulation cycle cannot propagate to the next (Paran, 2013).

Much like the PCD, we note that there is an entire range of values for Z_{DIM} where stability will be guaranteed. If $Z_{DIM} = 0$ then the magnitude of the transfer function cannot possibly be higher than 1 unless the distortions functions make it so, as Z_{ab} will always be more than 0. This will be true as well for the entire interval $Z_{HUT} > Z_{DIM} > 0$ as G_{ol} decreases in magnitude as the value of Z_{DIM} approaches the value Z_{HUT} .

$$\left| \frac{(Z_s \cdot Z_{HUT})}{(Z_{ab} + Z_{HUT})(Z_{ab} + Z_s)} \right| < 1 \quad (2.26)$$

We also note that if $Z_{DIM} < 2 \cdot Z_{HUT}$ a similar situation occurs.

$$\left| \frac{-(Z_s \cdot Z_{HUT})}{(Z_{ab} + Z_{HUT})(Z_{ab} + Z_s + 2 \cdot Z_{HUT})} \right| < 1 \quad (2.27)$$

The results above seem promising as unlike the ITM algorithm, stability is not entirely dependent on the impedance ratio of Z_s and Z_{HUT} , but instead, in the right choice of

Z_{DIM} . We also know that there is an entire range of values for Z_{DIM} where stability guaranteed, if we disregard the effects of the distortion functions. Finally, if $Z_{DIM} = Z_{HUT}$ the system cannot be unstable, no matter how long the delay or how bad the distortion. As such, if given the choice, Z_{DIM} should equal Z_{HUT} in any implementation of the DIM algorithm.

In reality though, precise knowledge of Z_{HUT} will rarely be available, meaning that it is important to look at the behaviour of the DIM IA when there mismatch between Z_{HUT} and Z_{DIM} . Figure 2.30 shows the Nyquist contours of the DIM algorithm for different values of Z_{DIM} in relation with Z_{HUT} . Like the PCD, due to the positive feedback present in the algorithm, we plot $-G_{ol}$, so that a direct comparison with the other algorithms can be made.

As predicted, the system was stable for all cases within the interval $2 \cdot Z_{HUT} > Z_{DIM} > 0$. In fact, the system remained stable until Z_{DIM} was four times larger than Z_{HUT} , showing that range of values of Z_{DIM} for which the system will be stable is large. Compared to the previous interface algorithms, the DIM IA performs extremely well in terms of stability, provided the choice of Z_{DIM} is reasonable. Figure 2.31 compares the DIM IA with the algorithms studied previously. We note that the stability margins are far greater than those of all previous methods, even when the error of Z_{DIM} is considerable, as seen in table 2.5.

TABLE 2.5. Stability Margins. DIM - System 1

	Gain margin	Phase margin
$Z_{DIM} = Z_{HUT}$	INF	INF
$Z_{DIM} = 0,8 \cdot Z_{HUT}$	17 dB	INF
$Z_{DIM} = 1,2 \cdot Z_{HUT}$	18.2 dB	INF
$Z_{DIM} = 0 \cdot Z_{HUT}$	0.13 dB	INF
$Z_{DIM} = 2 \cdot Z_{HUT}$	6.13 dB	INF
$Z_{DIM} = 4 \cdot Z_{HUT}$	0.101 dB	INF
$Z_{DIM} = 5 \cdot Z_{HUT}$	-1.05 dB	-177 °

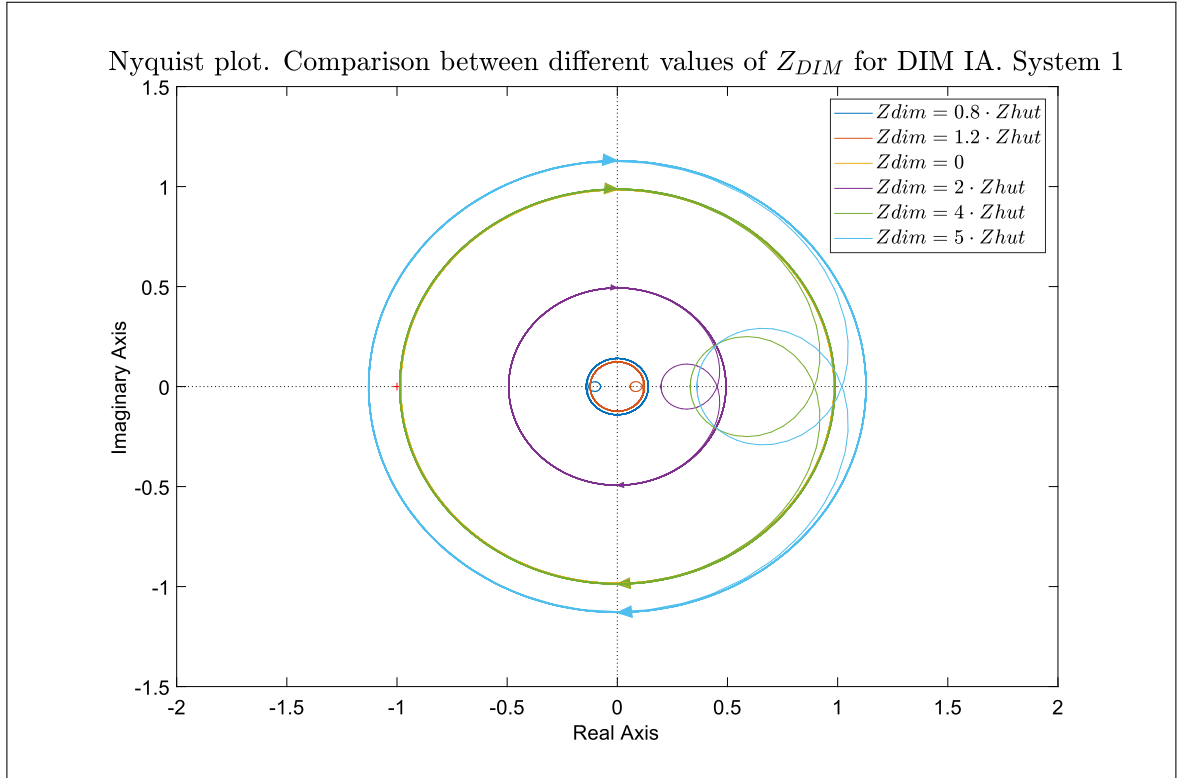


FIGURE 2.30. Nyquist contours of DIM algorithm. Multiple values for Z_{DIM}

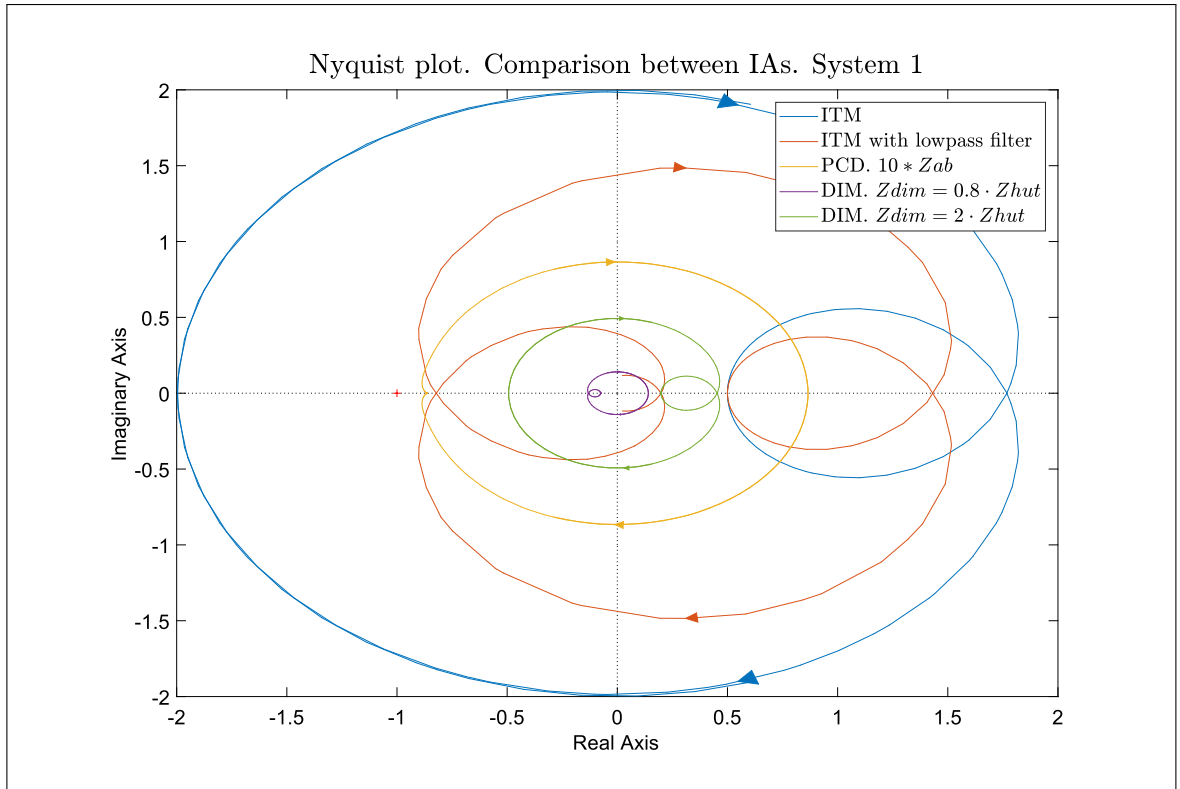


FIGURE 2.31. Comparison between the different interface algorithms

2.5.2. Accuracy considerations

For the DIM interface algorithm to be an ideal candidate for practical implementation it must not only be highly stable but also accurate. Like the with the previous IAs we will use its closed loop transfer function to study its behaviour. To calculate this function we will use the same procedure as for the PCD, shown in example 2.3.

Example 2.7. *We can calculate the closed loop transfer function as:*

$$G_{CL} = \frac{F \cdot G}{1 - G \cdot H} \quad (2.28)$$

Where F , G and H are defined as shown in example 2.6. Again, we make the assumption that $e^{st_b} = e^{st_c} = e^{st_e}$ and $G_a = G_b = G_e$, so that we can sum both feedback paths.

$$F = \frac{Z_{DIM}}{Z_s + Z_{ab} + Z_{DIM}} \quad (2.29)$$

$$G = \frac{G_a e^{st_a} \cdot Z_{HUT}}{Z_{ab} + Z_{HUT}} \quad (2.30)$$

$$\begin{aligned} H &= \frac{G_b e^{st_b} \cdot Z_s}{Z_s + Z_{ab} + Z_{DIM}} + \frac{-G_c e^{st_c} \cdot Z_s \cdot Z_{DIM}}{Z_{HUT} \cdot (Z_s + Z_{ab} + Z_{DIM})} \\ &= \frac{G_e e^{st_e} \cdot Z_s \cdot (Z_{HUT} - Z_{DIM})}{Z_{HUT} \cdot (Z_s + Z_{ab} + Z_{DIM})} \end{aligned} \quad (2.31)$$

With this, the closed loop transfer function of the DIM IA is as follows.

$$G_{CL} = \frac{Z_{HUT} \cdot Z_{DIM} \cdot G_a e^{-st_a}}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM}) - G_a \cdot G_e \cdot e^{-s(t_a+t_e)} Z_s (Z_{HUT} - Z_{DIM})} \quad (2.32)$$

In the previous section, we learned that the optimal value for Z_{DIM} in terms of stability was for it to be equal to Z_{HUT} . This meant that the system was stable regardless of

delay or distortion. As it turn out, this choice of damping impedance is not only optimal for stability but for accuracy as well .

Example 2.8. *Let us return to the closed loop transfer function in equation 2.32 and substitute $Z_{DIM} = Z_{HUT}$.*

$$G_{CL} = \frac{Z_{HUT} \cdot Z_{DIM} \cdot G_a e^{-st_a}}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM}) - G_a \cdot G_e \cdot e^{-s(t_a+t_e)} Z_s (Z_{HUT} - Z_{DIM})} \quad (2.33)$$

$$G_{CL} = \frac{Z_{HUT}^2 \cdot G_a e^{-st_a}}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT})} \quad (2.34)$$

From here, it is easy to see that if $Z_{ab} \approx 0$ the DIM algorithm will yield accurate results.

$$G_{CL} = \frac{Z_{HUT}^2 \cdot G_a e^{-st_a}}{(0 + Z_{HUT})(Z_s + 0 + Z_{HUT})} \quad (2.35)$$

$$G_{CL} = \frac{Z_{HUT} \cdot G_a e^{-st_a}}{Z_s + Z_{HUT}} \quad (2.36)$$

$$(2.37)$$

From example 2.8, it becomes clear that the choice $Z_{DIM} = Z_{HUT}$, yields the best results in terms of both, stability and accuracy. Figure 2.32 confirms this, as we see that the DIM method provides results which are almost identical to those of the NCS in terms of magnitude. The only difference between the two is the effect of the amplifier delay in the phase. One important observation is that the oscillations in the magnitude seen at higher frequencies in the previous IAs are not present.

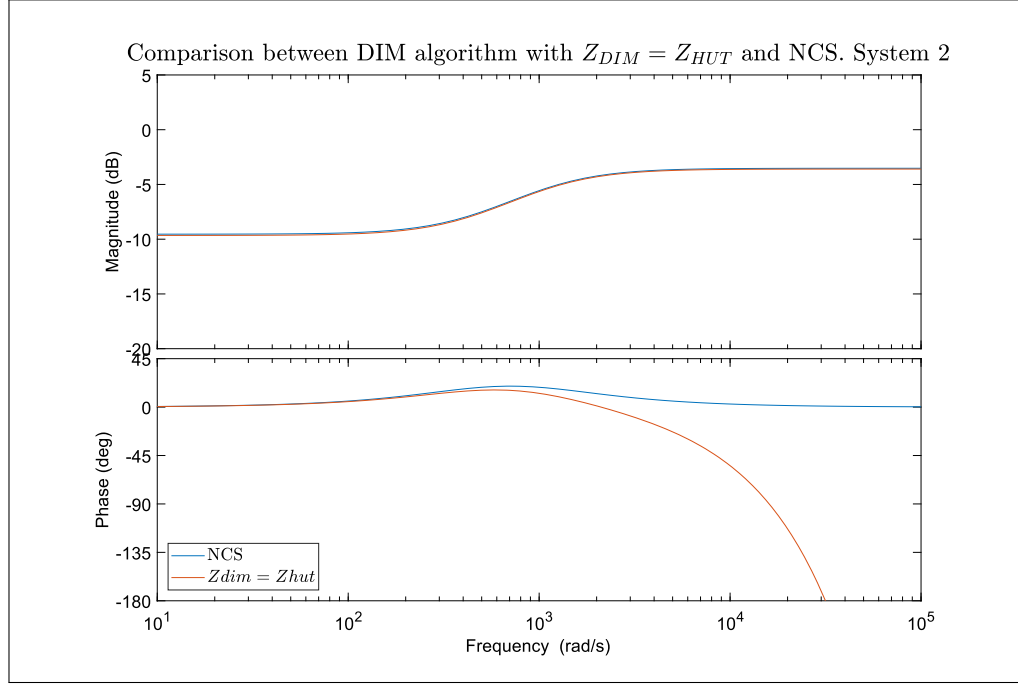


FIGURE 2.32. Comparison between the NCS and DIM

This is because when Z_{HUT} and Z_{DIM} are perfectly matched, the open loop transfer function is zero, meaning that the feedback path has no effect on the voltage V_{HUT} . This eliminates the effects internal delays have on the system, as seen as in the denominator of the transfer function where the term containing the delays is no longer present.

Nonetheless, if there is even a tiny mismatch between the impedances, these effect will reappear. Thus, it is still advised to filter the feedback signals in the same manner as with all the other IAs.

In practice, we are almost guaranteed to have a mismatch between Z_{HUT} and Z_{DIM} as perfect information about the impedance of the HUT is unlikely to be available. Most of the time, to implement the DIM IA we will either make a guess or try to obtain somehow a good estimate for Z_{HUT} to use as Z_{DIM} . Either way, our choice of Z_{DIM} will be subject to a certain degree of error meaning we must also understand how these error affect the accuracy of the IA. Figures 2.33 and 2.34 compare the NCS with systems using

the DIM IA with varying degrees of error in the choice of Z_{DIM} . Fortunately, we see that the DIM method is quite resilient to errors in the choice of Z_{DIM} as, for low frequencies, even poorly estimates produce accurate results. For higher frequencies, the correct choice of Z_{DIM} becomes much more important as accuracy is greatly reduced if even a small mismatch between the impedances is present.

Another important factor to achieve high accuracy is to reduce the value of Z_{ab} as much as possible as it will have a negative effect in the simulation's accuracy. Given that unlike the PCD, the stability of the simulation is not dependant on the value of this impedance, this is not a problem. In fact, in the DIM IA, Z_{ab} only represents the output impedance of the power amplifier which will generally be very low. Still, the effects of Z_{ab} are not negligible, as high large values for this impedance can seriously affect the results of a PHIL simulation, even if Z_{HUT} and Z_{DIM} are perfectly matched.

Despite the the above, the DIM algorithm has the potential for the best performance of all the IAs studied, seen in figures 2.36 and 2.37. This, as when Z_{DIM} is perfectly matched, we can even do away with some of the artefacts that affected the ITM, product of the internal delays. It also gets rid of the trade-off between stability and accuracy present, in all the other interface algorithms at the expense of increased complexity. This as we now must find an acceptable estimate for Z_{HUT} to use as Z_{DIM} to achieve an stable and accurate simulation. Still, the challenge of finding the right value for Z_{DIM} can be tackled in different ways, either by having prior knowledge about the HUT or by using an identification strategy to obtain a good estimate.

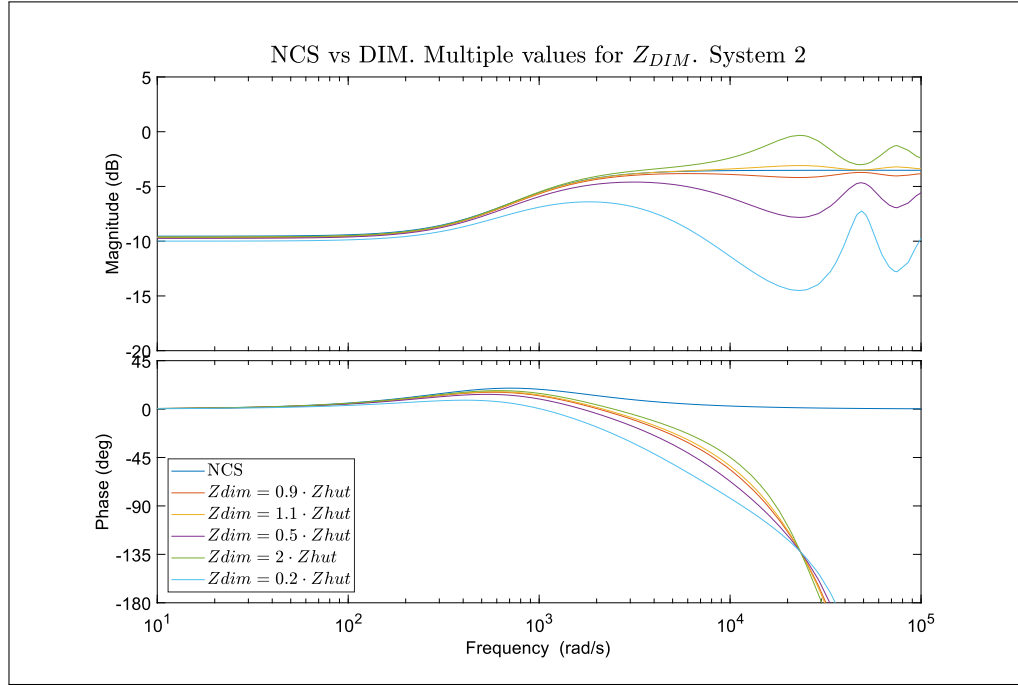


FIGURE 2.33. DIM. Multiple value of Z_{DIM}

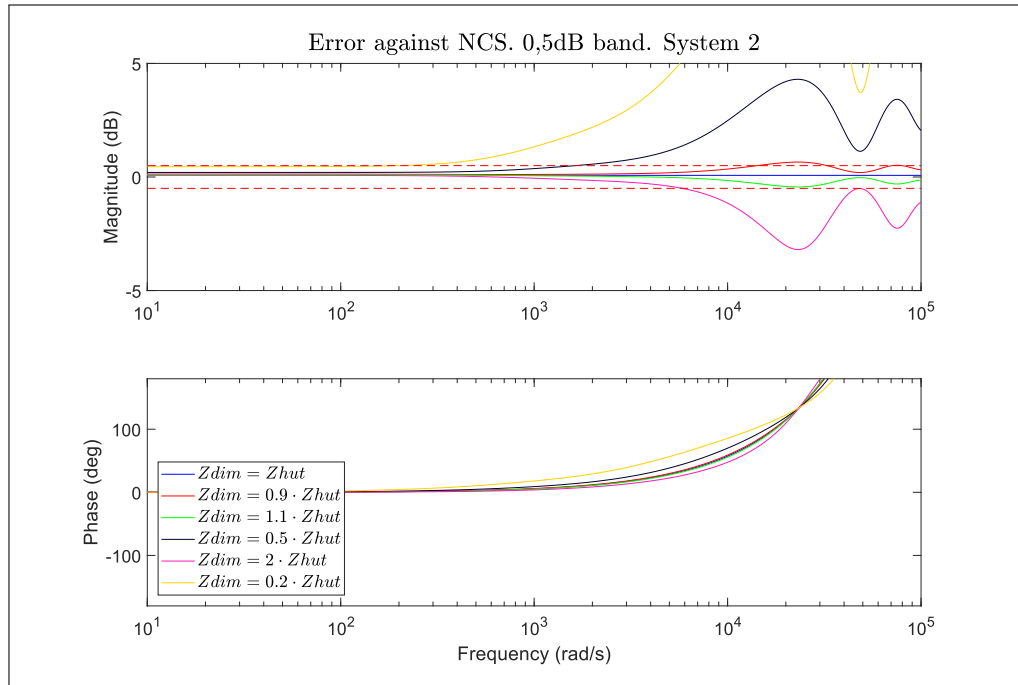


FIGURE 2.34. DIM. Multiple value of Z_{DIM} . Deviations with respect to NCS

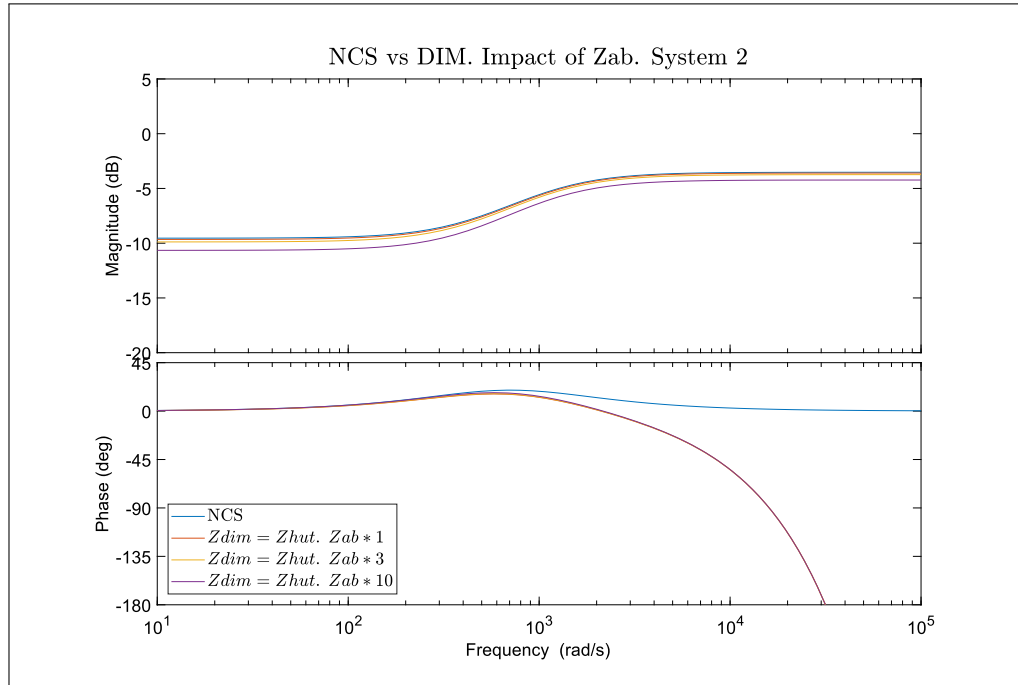


FIGURE 2.35. Comparison between the different values of Z_{ab}

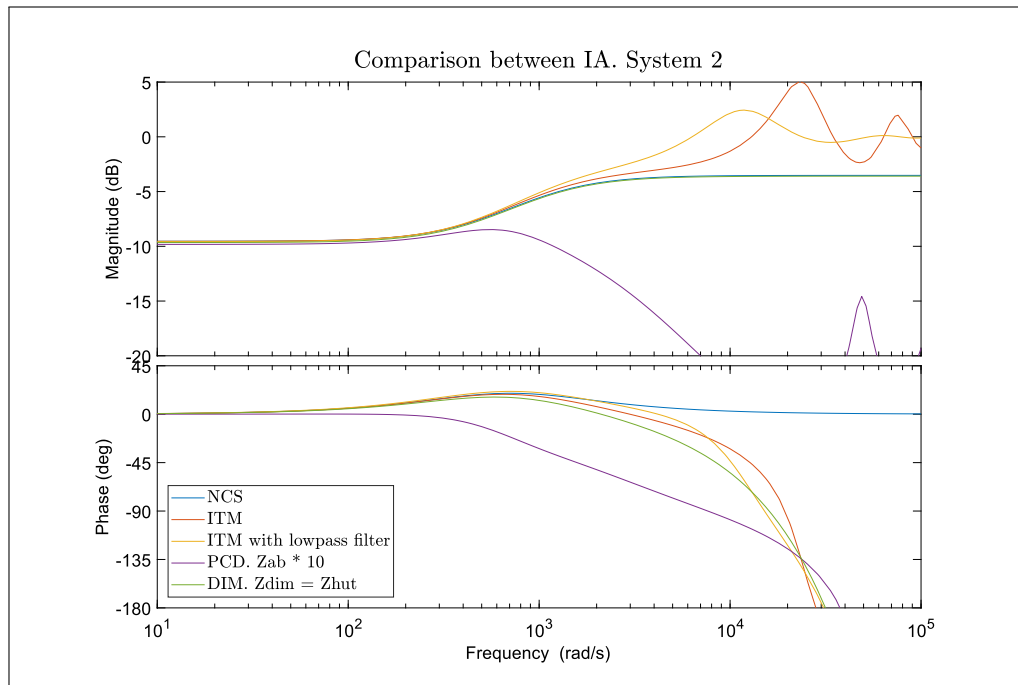


FIGURE 2.36. Comparison between different IAs

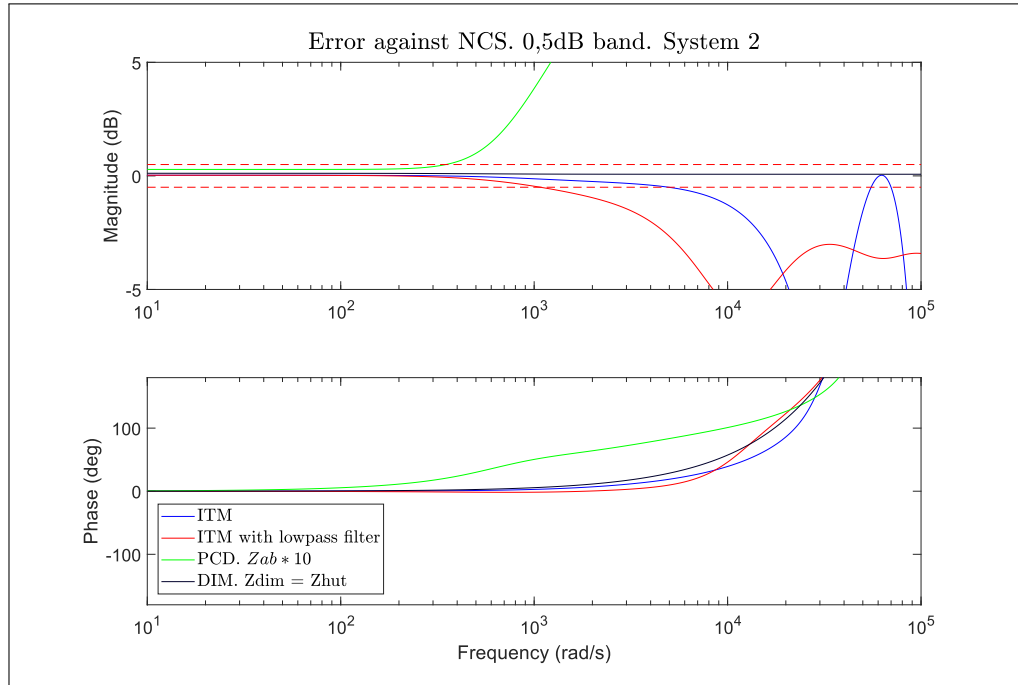


FIGURE 2.37. Comparison between different IAs. Deviation with respect to the NCS

2.6. Effects of an active HUT

So far we have analysed all interface algorithms representing our Hardware under test by the impedance, Z_{HUT} . This model is only valid for HUTs made out of passive devices, such as resistors, inductances or capacitors. Nonetheless, the most interesting applications of PHIL simulations have to do with active HUTs such as motors, inverters, rectifiers, etc. Thankfully, the analysis we have made so far can be easily extended to cover these types of HUTs.

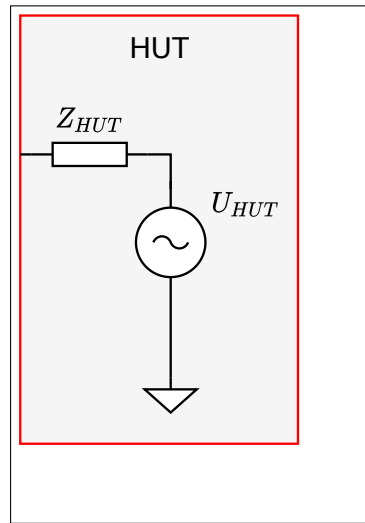


FIGURE 2.38. Active HUT

Instead of modelling our HUT with a just a single impedance, we will add a voltage source to the HUT. This voltage source will represent the active component of the HUT and it can have any behaviour. In the case of an inverter, this voltage source will represent the transistors as well as all the control logic of the device. The impedance Z_{HUT} will represent the output impedance of the device. In our inverter example, this would be the output filter. With this representation we can modify the control diagrams of the Interface algorithms to account for active HUTs. Figures 2.39 through 2.41 shown these modifications.

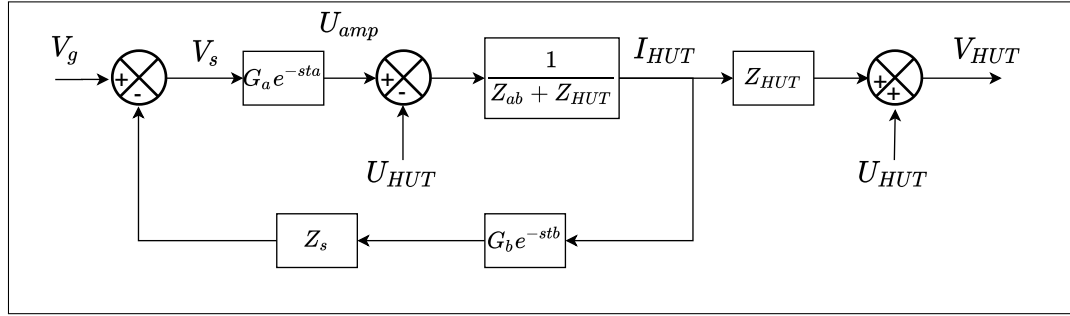


FIGURE 2.39. Block diagram of ITM IA with an active HUT

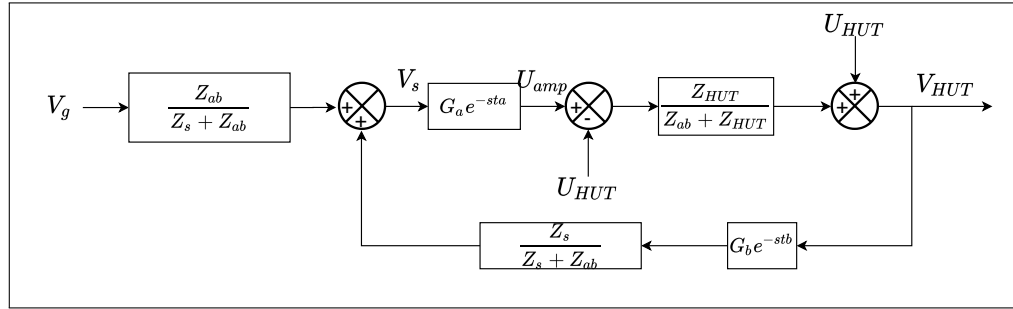


FIGURE 2.40. Block diagram of PCD IA with an active HUT

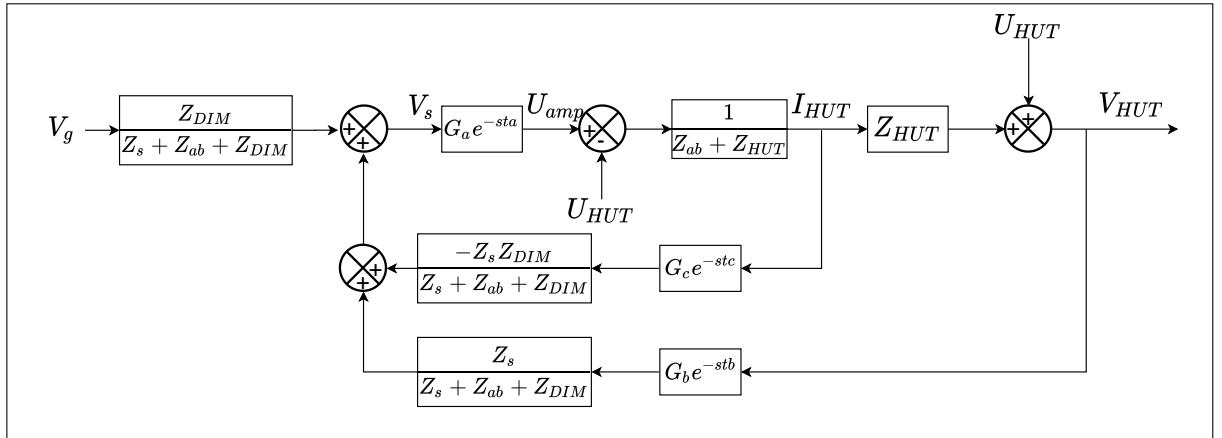


FIGURE 2.41. Block diagram of DIM IA with an active HUT

2.6.1. Effects on an active HUT on stability

A reasonable question that arises from the inclusion of an active HUT is whether it affects the stability of the PHIL simulation. We can answer this question by calculating the transfer function from U_{HUT} to V_{HUT} for the different IAs.

For the ITM this function is the following:

$$G_{ITM_{U-V}} = 1 - \frac{\frac{Z_{HUT}}{Z_{ab}+Z_{HUT}}}{1 + G_a G_b e^{-s(t_a+t_b)} \frac{Z_s}{Z_{ab}+Z_{HUT}}} \quad (2.38)$$

We notice that the denominator the G_{U-V} is nothing but $1 + G_{OL}$, meaning that the poles of this transfer function are the same as those of the closed loop transfer function shown in the previous section. As such, we can say that the addition of an active HUT makes no difference regarding the stability of the interface algorithm. In fact equation 2.38 can be simplified to:

$$G_{ITM_{(U-V)}} = 1 - \frac{\frac{Z_{HUT}}{Z_{ab}+Z_{HUT}}}{1 + G_{OL}} \quad (2.39)$$

A remark has to be made though, that the stability of the IA is not the same as the stability of the PHIL simulation as a whole. If the actions of the HUT were to turn the real system unstable, the same would happen in the PHIL simulation. That instability would not be a result of the IA but rather of the electrical dynamics of the simulated network.

A similar situation occurs with the PCD and DIM IAs. This should come as no surprise given the model that we have chosen for our active HUT. This, as all the complex dynamics that the HUT will have, stemming from its control systems for example, are lumped into the U_{HUT} and treated as a disturbance. The only part of the HUT that is relevant for the IA will be its electrical components represented in Z_{HUT} . For the PDC and DIM however, the influence of U_{HUT} over V_{HUT} is slightly different as they use the measured voltage as a feedback variable. For those, G_{U-V} is shown in equations 2.40 and

2.41. Nonetheless, the poles of the G_{U-V} remain the same as those of their respective IA's closed loop transfer functions.

$$G_{PCD(U-V)} = \frac{Z_{ab}(Z_{ab} + Z_s)}{(Z_{ab} + Z_s)(Z_{ab} + Z_{HUT}) - Z_s \cdot Z_{HUT} \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)}} \quad (2.40)$$

$$G_{DIM_{U-V}} = \frac{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM}) + Z_s Z_{HUT} \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)} - Z_{HUT}(Z_s + Z_{ab} + Z_{DIM})}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM}) - Z_s(Z_{HUT} - Z_{DIM}) \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)}} \quad (2.41)$$

2.6.2. Effects on an active HUT on accuracy

Another questions that arises with the inclusion of an active HUT is whether accuracy is preserved. To calculate the effects U_{HUT} has over V_{HUT} and to determined whether the different IAs produce accurate results we must first establish a baseline. As in the previous analysis this will be the response of the NCS, now modified to include an active HUT.

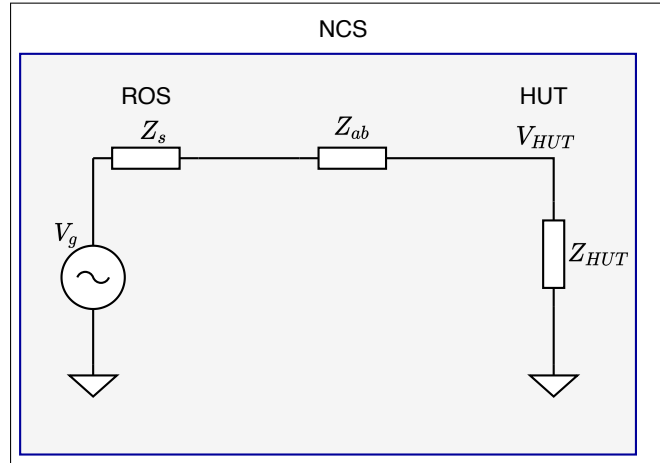


FIGURE 2.42. Naturally coupled system with an active HUT

From figure 2.42, it is easy to see that the correct value for the voltage V_{HUT} will be:

$$V_{HUT} = V_g \cdot \frac{Z_{HUT}}{Z_s + Z_{ab} + Z_{HUT}} + U_{HUT} \frac{Z_s + Z_{ab}}{Z_s + Z_{ab} + Z_{HUT}} \quad (2.42)$$

To see under which conditions accuracy is preserved, we will use the transfer functions presented in the previous section. For the ITM, the same conditions of low delay and distortion are necessary to maintain accuracy, in the presence of an active HUT.

$$G_{ITM_{U-V}} = 1 - \frac{\frac{Z_{HUT}}{Z_{ab} + Z_{HUT}}}{1 + G_a G_b e^{-s(t_a + t_b)} \frac{Z_s}{Z_{ab} + Z_{HUT}}} \quad (2.43)$$

$$= 1 - \frac{Z_{HUT}}{Z_{ab} + Z_{HUT} + Z_s \cdot G_a G_b e^{-s(t_a + t_b)}} \quad (2.44)$$

$$= \frac{Z_{ab} + Z_s \cdot G_a G_b e^{-s(t_a + t_b)}}{Z_{ab} + Z_{HUT} + Z_s \cdot G_a G_b e^{-s(t_a + t_b)}} \quad (2.45)$$

Equation 2.45 shows as well that same problems caused by the internal delays will be present. They can however, still be mitigated with the inclusion of a low pass filter in the feedback path as seen in equation 2.46.

$$G_{LPF_{U-V}} = \frac{Z_{ab} + Z_s \cdot G_a G_b e^{-s(t_a + t_b)}}{Z_{ab} + Z_{HUT} + Z_s \cdot G_a G_b G_{LPF} e^{-s(t_a + t_b)}} \quad (2.46)$$

For the PCD, the situation is similar. We note in equation 2.48 that should $G_a \cdot G_b \cdot e^{-s(t_a + t_b)} \approx 1$, we can obtain the correct result. We note that unlike the other algorithms, in the PCD Z_{ab} is not just the output impedance of the power amplifier but an added physical component whose values is critical for the stability of the simulation. Just like in the normal case, high values will compromise accuracy.

$$G_{PCD_{(U-V)}} = \frac{Z_{ab}(Z_{ab} + Z_s)}{(Z_{ab} + Z_s)(Z_{ab} + Z_{HUT}) - Z_s \cdot Z_{HUT} \cdot G_a \cdot G_b \cdot e^{-s(t_a + t_b)}} \quad (2.47)$$

$$= \frac{Z_{ab}(Z_{ab} + Z_s)}{Z_{ab}^2 + Z_s Z_{ab} + Z_{HUT} Z_{ab} + Z_s Z_{HUT} - Z_s Z_{HUT}} \quad (2.48)$$

$$= \frac{Z_{ab} + Z_s}{Z_s + Z_{ab} + Z_{HUT}} \quad (2.49)$$

Finally, the impacts of an active HUT on the DIM IA are perhaps the most interesting. The IA remains accurate given the right conditions but unlike in the passive case, $Z_{DIM} = Z_{HUT}$ does not completely eliminate the effects of the feedback loops. While a perfect matching of Z_{DIM} and Z_{HUT} is still able to eliminate the effects of the internal delays, it does not remove the distortion and delay of the sensors from the equation, like in the passive case. Applying $Z_{DIM} = Z_{HUT}$ to equation 2.50 still leaves $G_a \cdot G_b \cdot e^{-s(t_a+t_b)}$ in the numerator. Only when we make $G_a \cdot G_b \cdot e^{-s(t_a+t_b)} \approx 1$ in equation 2.51, we can begin to obtain the correct answer.

$$G_{DIMU-V} = \frac{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM}) + Z_s Z_{HUT} \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)} - Z_{HUT}(Z_s + Z_{ab} + Z_{DIM})}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{DIM}) - Z_s(Z_{HUT} - Z_{DIM}) \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)}} \quad (2.50)$$

$$= \frac{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT}) + Z_s Z_{HUT} \cdot G_a \cdot G_b \cdot e^{-s(t_a+t_b)} - Z_{HUT}(Z_s + Z_{ab} + Z_{HUT})}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT})} \quad (2.51)$$

$$= \frac{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT}) + Z_s Z_{HUT} - Z_{HUT}(Z_s + Z_{ab} + Z_{HUT})}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT})} \quad (2.52)$$

$$= \frac{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT}) + Z_s Z_{HUT} - Z_{HUT} \cdot Z_s - Z_{HUT} \cdot Z_{ab} - Z_{HUT}^2}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT})} \quad (2.53)$$

$$= \frac{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT}) - Z_{HUT}(Z_{ab} + Z_{HUT})}{(Z_{ab} + Z_{HUT})(Z_s + Z_{ab} + Z_{HUT})} \quad (2.54)$$

$$= \frac{Z_s + Z_{ab}}{Z_s + Z_{ab} + Z_{HUT}} \quad (2.55)$$

In summary, adding an active HUT does not significantly alter the properties of each IA, meaning these should remain both stable and accurate under the same conditions for passive and active HUTs alike.

2.7. Chapter conclusion

Throughout this chapter we have described the different interface algorithms proposed in the literature and analysed their properties in relation to accuracy and stability. We have also discussed what are the challenges of implementing each IA. Tables 2.6 and 2.7 synthesise the findings of this chapter, comparing the IAs across several dimensions of interest.

TABLE 2.6. Interface algorithms: Summary

Characteristic	ITM	ITM w LPF	PCD	DIM
Stability	Low	Moderate	Best	Very High
Accuracy	Best	High	Low	Very High
Complexity	Low	Low	Low	High
Added physical components	No	No	Yes	No
Sensors	Current	Current	Voltage	Voltage & Current

TABLE 2.7. Interface algorithms: Open-loop transfer functions

Interface algorithm	Open-loop transfer function
ITM	$-G_a \cdot G_b e^{-s(t_a+t_b)} \frac{Z_s}{Z_{ab}+Z_{HUT}}$
ITM w LPF	$-G_{LPF} \cdot G_a \cdot G_b e^{-s(t_a+t_b)} \frac{Z_s}{Z_{ab}+Z_{HUT}}$
PCD	$G_a e^{-st_a} \cdot G_b e^{-st_b} \frac{Z_s \cdot Z_{HUT}}{(Z_{ab}+Z_s)(Z_{ab}+Z_{HUT})}$
DIM	$G_e e^{st_e} \cdot G_a e^{st_a} \frac{Z_s \cdot (Z_{HUT}-Z_{DIM})}{(Z_{ab}+Z_{HUT})(Z_s+Z_{ab}+Z_{DIM})}$

Much like previous works, we find that the damping impedance method provides the best performance of all the IAs, allowing us to not have to compromise between the stability and accuracy of a simulation. This improved performance comes at the cost of ease of implementation as we must now find a way to estimate Z_{HUT} . Having considered all the facts, we provide recommendations about when to use each interface algorithm.

If it is known that the impedances of the simulated network and the HUT will not form an unstable combination, we advice the use of the ITM. This, as it is the simplest IA to implement and will result in an accurate simulation provided the delays and distortions are low. We advice as well, the addition of a lowpass filter on the current feedback to mitigate the effect of the internal delays. If it is not certain that Z_s and Z_{HUT} form an stable pair, one may still try to verify this experimentally, as using the ITM vs the DIM IA for example, may save much valuable time and resources. This however, should only be attempted if both the HUT and amplifier are properly protected in the case the experiment results in instability.

Should Z_s and Z_{HUT} form a unstable pair for the ITM, one may attempt to stabilise the simulation by lowering the cut-off frequency of the lowpass filter. This is simple to implement and test, and may prove very cost effective. This should only be done, if the phenomena one wishes to study are all within the bandwidth of the lowpass filter. If this is not the case, the results will prove inaccurate.

Finally, if neither of the approaches presented above results in a satisfactory simulation, one must use DIM as it allows for an stable and accurate simulation regardless of the values of Z_s and Z_{HUT} . This of course presents a new challenge: the estimation of Z_{HUT} . Given that our goal is to implement a PHIL platform that is able to perform the widest possible range of simulations, the DIM IA is the logical choice. This means that we must design and implement a procedure that allows us to obtain a good guess of the value of Z_{HUT} , for any device we wish to test. This is precisely the path we have chosen and is the content of the next few chapters.

3. IMPEDANCE IDENTIFICATION

3.1. Introduction

As was established in Chapter 2, the best performing interface algorithm for PHIL simulations is the DIM, however its implementation presents the challenge of identifying Z_{HUT} . Given that our goal is to implement a PHIL platform capable of a wide range of experiments, the ITM is not a suitable option, given its stability problems. This means that we must implement a way of finding a good estimate of the HUTs impedance, to perform our PHIL simulations.

Ideally any implemented identification routine must run concurrently with the execution of the electrical simulation. Achieving this can be challenging, as the electrical simulation must be run with a very small time step, potentially limiting the identification techniques that can be used. This, as complex and computationally heavy identification algorithms may not execute fast enough to not interfere with the simulation.

To get around this, as it will be explained in Chapter 4, we took advantage of the multiple cores in the PHIL simulation target allocating one core to exclusively run the identification algorithm. It was also possible to get the cores to run with different time steps. This allowed the electrical simulation to run with a very small time step, thus not compromising simulation accuracy, while giving ample time in other core to run a rather complex identification routine.

The identification routine selected for this application performs first a non-parametric identification technique known as spectral analysis followed by parameter fitting process.

Spectral analysis was selected as a first step as it allows to identify the impedance over large range of frequencies allowing to better characterise the HUT's impedance. The use of this technique has been proposed by several researchers (Sieggers y Santi, 2014) (Liegmann et al., 2016) and has been used successfully in PHIL platforms that use the DIM IA, such as that implemented by Riccobono et al. (2017). After spectral analysis,

parameters are fitted to the gathered data, using the Levenberg-Marquardt algorithm to produce the final impedance model.

Many previous works use a very simpler strategy, measuring only the magnitude and the phase of the voltage and current to estimate the resistance and inductance of the HUT (Paran y Edrington, 2013),(Jiang et al., 2019). While simple, this will only find a good estimate for Z_{HUT} at the fundamental frequency but may miss important, high frequency features of the devices impedance. Instead, spectral analysis captures much the information about the HUT's impedance over a broad frequency spectrum, allowing for a more accurate simulation.

It is one of our hypothesis that together, these two techniques will allow for an identification routine that successfully identifies the HUT's impedance on a timely manner and with the precision required by the DIM interface algorithm. In this chapter, the procedure and the mathematical tools to perform the identification process will be explored with a emphasis on their practical implementation.

3.2. Identification procedure overview

Non-parametric identification is often used when one has little to no knowledge about the system at hand. Given that our goal is to obtain an identification routine that is able to deal with the widest range of HUT's possible and that we are unlikely to have prior knowledge of the design or construction of the HUT, non-parametric identification was chosen to as the first step in our identification routine.

Given that our goal is to obtain an approximation of the transfer function of the impedance of the HUT spectral analysis was chosen among many non-parametric identification techniques as it presents a straightforward procedure to obtain such information. This process has the following steps:

1. Excite the system with an appropriate input
2. Measure the systems input and output

3. Perform a correlation analysis with the measured input and output
4. Take the Discrete Fourier transform of the quantities in the previous step to obtain an estimate of their power spectral densities (PSD)
5. Divide both power spectral densities to obtain an estimate of the frequency response of the system.

Once an empiric estimate for the frequency response of the impedance has been found, we apply the levenberg-Marquardt algorithm to get a parametric model of the impedance based on the generated data.

3.3. Data gathering

Given the nature of our PHIL platform, the impedance of the HUT must be identified while the rest of the simulation is ongoing and be re-identified during the simulation at fixed intervals. This means that our impedance will not just be subject to our identification input, but also to the signals coming from the rest of the simulation. In fact, these signals will be, with all certainty, of a much greater amplitude than the identification input itself. Added to this, we will have noise generated by the amplifier and the current sensors. Thus, our system can be described by the following equation 3.1.

$$y(t) = u(t)G(q) + e(t)H(q) \quad (3.1)$$

Here, $u(t)$ will be the sum of our selected identification input $id(t)$, in this case, white noise, and the signals coming from the rest of the simulation $s(t)$. $G(q)$ the system that we wish to identify while $e(t)$ and $H(q)$ represent disturbances related to the process and the way these disturbances influence the output.

The identification environment can be better understood with the aid of figure 3.1. We can see how $u(t)$ is given as the amplifiers input to form the voltage that is applied to the HUT, V_{HUT} . Afterwards, the current of the HUT is measured by a current sensor. This

final signal will be our identification output $y(t)$, thus by studying it and its relation with $u(t)$ we aim to identify $G(q)$.

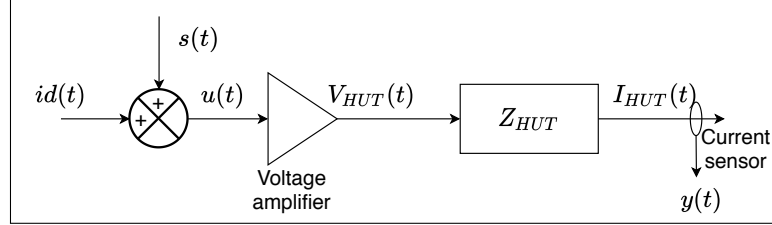


FIGURE 3.1. Data gathering environment

3.4. Correlation functions and power spectral density

Spectral analysis relies on obtaining an estimate for the power spectral density of the input and output of a process by exploiting its relationship with their corresponding correlation functions. As such, it will be helpful to introduce all these concepts. Further reading about these concepts can be found in Chapters 1 and 2 of *Spectral Analysis of Signals* by [Stoica y Moses \(2005\)](#) and Chapter 3 of *System identification* [Torsten y Stoica \(1989c\)](#) . Equations 3.2 and 3.3 in definition 3.1 introduce the auto correlation and cross correlation functions.

Definition 3.1. *The autocorrelation function of a signal $x(t)$ is given by*

$$r_{xx}(\tau) := E[u(t + \tau)u(t)] \quad (3.2)$$

while the cross correlation between two signals $y(t)$ and $x(t)$ is given by

$$r_{yx}(\tau) := E[y(t + \tau)u(t)] \quad (3.3)$$

Where $E[x]$ is the expected value operator.

While these are the rigorous definitions of the correlation functions we require, they are not very helpful in terms of empirically calculating them. Nonetheless, these quantities can be estimated as seen in equations 3.4 and 3.5, in definition 3.2.

Definition 3.2. *For a real, discrete signal $x[n]$ of length N , its autocorrelation function can be estimated as*

$$\hat{r}_{xx}(\tau) := \frac{1}{N} \sum_{k=1}^N x[k]x[k + \tau], \quad |\tau| \leq N - 1 \quad (3.4)$$

The cross correlation function between two real, discrete signals $y[k]$ and $x[k]$, both of length N can be estimated by

$$\hat{r}_{yx}(\tau) := \frac{1}{N} \sum_{k=1}^N y[k]x[k + \tau], \quad |\tau| \leq N - 1 \quad (3.5)$$

We will also introduce the concept of power spectral density, periodogram and its relation with the correlation functions. These are important as the PSD can act as a proxy for the spectral content of a given quantity and can be calculated from the correlation functions by the Wiener-Khinchin theorem.

Theorem 3.1. *Given a discrete time series $x[n]$, the power spectral density $\Phi_x(\omega)$ is*

$$\Phi_x(\omega) = \sum_{\tau=-\infty}^{\infty} r_{xx}[\tau]e^{-i\omega\tau} \quad (3.6)$$

Where $r_{xx}[\tau]$ is the autocorrelation function of $x[n]$.

This tells us that the PSD of a signal can be obtained by taking the Fourier transform of its correlation function. Furthermore, the inverse is also true, taking the inverse Fourier transform of the PSD will grant us the respective correlation function. This one-to-one correspondence means that the correlation function and the PSD hold, in essence, the same information.

Theorem 3.2. *Given the power spectral density $\Phi_x(\omega)$, its autocorrelation function is given by:*

$$r_{xx}[\tau] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \Phi_x(\omega) e^{i\omega\tau} d\omega \quad (3.7)$$

The periodogram is an estimate of the PSD that converges to the actual value when T (and in consequence N , to maintain Δt) approach infinity. This method for estimating the PSD can usually be computed more efficiently as it just needs to calculate the Fourier transform of the data, which is done through the FFT algorithm. Nonetheless, using the periodogram is not without drawbacks, as seen in further sections.

Definition 3.3. *The periodogram of a real valued sequence $x[n]$ of length N , period T and sample time $\Delta t = T/N$ is given by:*

$$U_n(\omega) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] e^{-i2\pi\omega\Delta t} \quad (3.8)$$

$$\hat{\Phi}_x(\omega)^P = \frac{1}{N} |U_n(\omega)|^2 \quad (3.9)$$

We can also define a periodogram to estimate the cross power spectral density between two signals $x[n]$ and $y[n]$.

$$Y_n(\omega) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} y[n] e^{-i2\pi\omega\Delta t} \quad (3.10)$$

$$\hat{\Phi}_{yx}(\omega)^P = \frac{1}{N} Y_n(\omega) U_n(\omega)^* \quad (3.11)$$

3.5. Spectral analysis

Spectral analysis aims to obtain an estimate of the transfer function of a system based on estimations of the PSD of the systems inputs and outputs. Given our data generating model described in 3.1, equations 3.12 and 3.13 hold true. These two equations relate the estimated PSD's with the transfer functions $G(e^{j\omega})$ and $H(e^{j\omega})$.

$$\Phi_y(\omega) = |G(e^{j\omega})|^2 \Phi_u(\omega) + \sigma^2 |H(e^{j\omega})|^2 \quad (3.12)$$

$$\Phi_{yu}(\omega) = G(e^{j\omega}) \Phi_u(\omega) \quad (3.13)$$

Equation 3.13 tells us allows us that $G(e^{j\omega})$ can be calculated by dividing $\Phi_{yu}(\omega)$ by $\Phi_u(\omega)$. Thus an estimate of the plant can be obtained with the estimates of this two quantities. Let us verify equations 3.12 , 3.13.

Example 3.1. *Given our data gathering model in equation 3.1 where: $G(q)$ and $H(q)$ represent the dynamics of the input and perturbations as shown in equation 3.14. q is the shift operator; while $g(k)$ and $h(k)$ are the respective weighting functions of $G(q)$ and $H(q)$ in the time domain.*

$$G(q) = \sum_{k=0}^{\infty} g(k)q^{-k} \quad H(q) = \sum_{k=0}^{\infty} h(k)q^{-k} \quad (3.14)$$

Assuming $u(t)$ is an stationary stochastic process and it is uncorrelated to the disturbance $e(t)$, let us calculate Φ_y to try to prove equation 3.12.

First we will calculate the autocorrelation of $y(t)$.

$$r_y = Ey(t + \tau)y^T(t) \quad (3.15)$$

$$= \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} g(j)Eu(t + \tau - j)u^T(t - \tau)g^T(k) + \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} h(j)Ee(t + \tau - j)e^T(t - \tau)h^T(k) \quad (3.16)$$

$$= \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} g(j)r_u(\tau - j + k)g^T(k) + \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} h(j)r_e(\tau - j + k)h^T(k) \quad (3.17)$$

Now we can take the discrete Fourier transform to obtain Φ_y

$$\Phi_y = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} g(j)e^{-ij\omega}r_u(\tau - j + k)e^{-i(\tau-j+k)\omega}g^T(k)e^{ik\omega} + \sum_{\tau=-\infty}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} h(j)e^{-ij\omega}r_e(\tau - j + k)e^{-i(\tau-j+\frac{1}{2\pi}k)\omega}h^T(k)e^{ik\omega} \quad (3.18)$$

$$= \frac{1}{2\pi} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} g(j)e^{-ij\omega} \left[\sum_{\tau'=-\infty}^{\infty} r_u(\tau')e^{-i\tau'\omega} \right] g^T(k)e^{ik\omega} + \frac{1}{2\pi} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} h(j)e^{-ij\omega} \left[\sum_{\tau'=-\infty}^{\infty} r_e(\tau')e^{-i\tau'\omega} \right] h^T(k)e^{ik\omega} \quad (3.19)$$

$$= \left[\sum_{j=0}^{\infty} g(j)e^{-ij\omega} \right] \Phi_u \left[\sum_{k=0}^{\infty} g^T(k)e^{ik\omega} \right] + \left[\sum_{j=0}^{\infty} h(j)e^{-ij\omega} \right] \Phi_e \left[\sum_{k=0}^{\infty} h^T(k)e^{ik\omega} \right] \quad (3.20)$$

$$= G(e^{-i\omega})\Phi_u G(e^{-i\omega})^T + H(e^{-i\omega})\Phi_e H(e^{-i\omega})^T \quad (3.21)$$

$$= |G(e^{-i\omega})|^2 \Phi_u + \sigma^2 |H(e^{-i\omega})|^2 \quad (3.22)$$

The proof for equation 3.13 follows the same logic. We first calculate the crosscorrelation between $u(t)$ and $y(t)$.

$$r_y u = E y(t + \tau) u^T(t) \quad (3.23)$$

$$= \sum_{j=0}^{\infty} g(j) E u(\tau - j) u^T(t - \tau) \quad (3.24)$$

$$= \sum_{j=0}^{\infty} g(j) r_u(\tau - j) \quad (3.25)$$

Again, taking the Fourier we obtain Φ_{yu} .

$$\Phi_{yu} = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} \sum_{j=0}^{\infty} g(j) e^{-ij\omega} r_u(\tau - j) e^{-i(\tau-j)\omega} \quad (3.26)$$

$$= \sum_{j=0}^{\infty} g(j) e^{-ij\omega} \left[\frac{1}{2\pi} \sum_{\tau'=-\infty}^{\infty} r_u(\tau') e^{-i(\tau')\omega} \right] \quad (3.27)$$

$$= G(e^{-i\omega}) \Phi_u \quad (3.28)$$

Example 3.1 shows equation 3.13 holds true, meaning we can obtain the frequency response of system $G(e^{j\omega})$ by dividing $\Phi_{yu}(\omega)$ by $\Phi_u(\omega)$. This in turn, means that we can obtain an estimate of $\hat{G}(e^{j\omega})$ with equation 3.33. Given real signals $y[k]$ and $u[k]$, $\hat{\Phi}_{yu}(\omega)$ and $\hat{\Phi}_u(\omega)$ can be calculated as follows:

$$\hat{r}_u(\tau) := \frac{1}{N} \sum_{k=1}^N u[k] u[k - \tau] \quad (3.29)$$

$$\hat{r}_{yu}(\tau) := \frac{1}{N} \sum_{k=1}^N y[k] u[k - \tau] \quad (3.30)$$

$$\hat{\Phi}_u(\omega) = \sum_{\tau=-N}^N \hat{r}_u(\tau) e^{-i\omega\tau} \quad (3.31)$$

$$\hat{\Phi}_{yu}(\omega) = \sum_{\tau=-N}^N \hat{r}_{yu}(\tau) e^{-i\omega\tau} \quad (3.32)$$

$$\hat{G}(e^{j\omega}) = \frac{\hat{\Phi}_{yu}(\omega)}{\hat{\Phi}_u(\omega)} \quad (3.33)$$

Alternatively, we may also try using the periodogram to calculate estimates for the spectrum $\hat{\Phi}_u$ and $\hat{\Phi}_{yu}$ directly using equations 3.9 and 3.11, as $Y_n(\omega)$ and $U_n(\omega)$ can be calculated efficiently through the FFT algorithm.

$$\hat{\Phi}_u(\omega)^P = \frac{1}{N} |U_n(\omega)|^2 \quad (3.34)$$

$$\hat{\Phi}_{yu}(\omega)^P = \frac{1}{N} Y_n(\omega) U_n(\omega)^* \quad (3.35)$$

3.6. Windowing

The process laid out so far will often produce poor results. In the case where $u(t)$ is an stochastic process, then the estimates for the spectrum will not converge to the true spectrums as the number of samples goes to infinity. In particular $\hat{\Phi}_{yu}$ will on average behave like Φ_{yu} , but its variance will not tend to zero. For further reading about the statistical properties of the periodogram, see (Brillinger, 1981). One of the reasons for this is that the estimate for $\hat{r}_{yu}(\tau)$ will be inaccurate for large values of τ . Given that our estimate for $\hat{\Phi}_{yu}$ in equation 3.32, weights all values of $\hat{r}_{yu}(\tau)$ equally, meaning the whole estimation will lose accuracy. A similar analysis holds true for $\hat{r}_u(\tau)$ and $\hat{\Phi}_u$.

Given that the poorly estimated values of the correlations functions $\hat{r}_{yu}(\tau)$ and $\hat{r}_u(\tau)$ when τ is large are the cause of the inaccuracies, the problem may be resolved if these

values are given less weight in, while values where τ is closer to zero are given more importance. This can be done via the application of a lag window $w(\tau)$.

A lag window is a function by which we will multiply our computed correlation sequence. It should be 1 for $\tau = 0$ and decrease as τ increases, being 0 for large values of τ . We will call the length of this window γ .

Several windowing functions have been proposed in the literature with the most common being the rectangular function, seen in $w_1(\tau)$; The Hann function, $w_2(\tau)$; and Hamming function, $w_3(\tau)$. For our application the Hann window was selected as it has been tried and tested in applications similar to ours.

Example 3.2. *Some common windowing functions*

$$w_1(\tau) = \begin{cases} 1 & |\tau| \leq \gamma \\ 0 & |\tau| > \gamma \end{cases} \quad (3.36)$$

$$w_2(\tau) = \begin{cases} \frac{1}{2}(1 + \cos(\frac{\pi\tau}{\gamma})) & |\tau| \leq \gamma \\ 0 & |\tau| > \gamma \end{cases} \quad (3.37)$$

$$w_3(\tau) = \begin{cases} 0,5383 - 0,4616(\cos(\frac{\pi\tau}{\gamma})) & |\tau| \leq \gamma \\ 0 & |\tau| > \gamma \end{cases} \quad (3.38)$$

With the application of a windowing function, equations 3.31 and 3.32 become, effectively yielding the practical equations that will be used to perform the spectral estimation. This method is known as the Blackman-Tukey method for spectral estimation (Blackman y Tukey, 1958).

$$\hat{\Phi}_{yu}(\omega) = \sum_{\tau=-\gamma}^{\gamma} w(\tau) \hat{r}_{yu}(\tau) e^{-i\omega\tau} \quad (3.39)$$

$$\hat{\Phi}_u(\omega) = \sum_{\tau=-\gamma}^{\gamma} w(\tau) \hat{r}_u(\tau) e^{-i\omega\tau} \quad (3.40)$$

The regardless of the chosen windowing function, the length γ of said functions will have an important effect on the identification process. Choosing γ too large will not provide the desired effect to reduce the influence of the poorly identified values of the correlation functions for large τ . Choosing gamma too small and the spectrum will be smoothed, potentially getting rid of important sharp peaks and reducing frequency resolution of the identification. The choice of γ is not straightforward however literature recommends choosing between 5 % to 10 % of the number of samples and see if whether the results contain all the detail that is needed for the identification. If not, γ should be increased until the results are satisfactory. (Ljung, 1999, pp. 168–197)

Windowing techniques can also be implemented to obtain better spectral estimates with the periodogram method, in fact this is the basis of the Welch method for spectral estimation (Welch, 1967). Nonetheless, this method can be harder to implement and the estimates it provides are not statistically better than the ones provided by the Blackman-Tukey method. Still it remains popular and viable option for spectral estimation.

3.7. Excitation

In order to identify the frequency response of a given system, it is necessary to excite said system with a known signal. However the properties of this signal are relevant to the identification process and hence it must be adequately selected.

3.7.1. Sine and sum of sines

Given that the aim of this process is to identify the frequency response of system, using a sinusoidal signal may seem as a reasonable option. As we know, when excited with a sinusoidal input a LTI system it will start to oscillate with a certain magnitude and phase with respect to the input. In theory, if we took several sinusoidal inputs of different frequencies and applied them one by one to the system, we could calculate the magnitude and phase of each point and thus reconstruct the frequency response. This is indeed a valid identification technique however it is very time inefficient requiring many separate test to

obtain the information needed. It is also not possible to apply this technique in an online scenario thus not resulting practical for this application.

3.7.2. Step function

Step functions -by contrast- have a much wider spectral content as it can be seen in figure 3.2. In fact, in steps most of the energy is concentrated in the lower frequencies. This is generally useful as lower frequency poles tend to dominate the dynamics of real systems making them specially important for identification purposes. The step response is also an important dynamic in and of it self for most systems, as control inputs are often steps or series of steps. From this response, useful information can be easily obtained, quantities such as: rise time, settling time, overshoot as well as potential resonance frequencies can all be identified with this method. Because of this, steps are common identification inputs for simple systems. However, a step function is not a suitable identification input for our application. In a PHIL platform the identification routine must be run while the rest of the simulation is running without affecting the validity of the results. Applying a step of sufficient magnitude to properly identify the HUT's impedance, Z_{HUT} , would undoubtedly alter the simulation in a significant way. Furthermore, identification inputs should ideally be persistent over time in order to properly identify systems with complex dynamics.

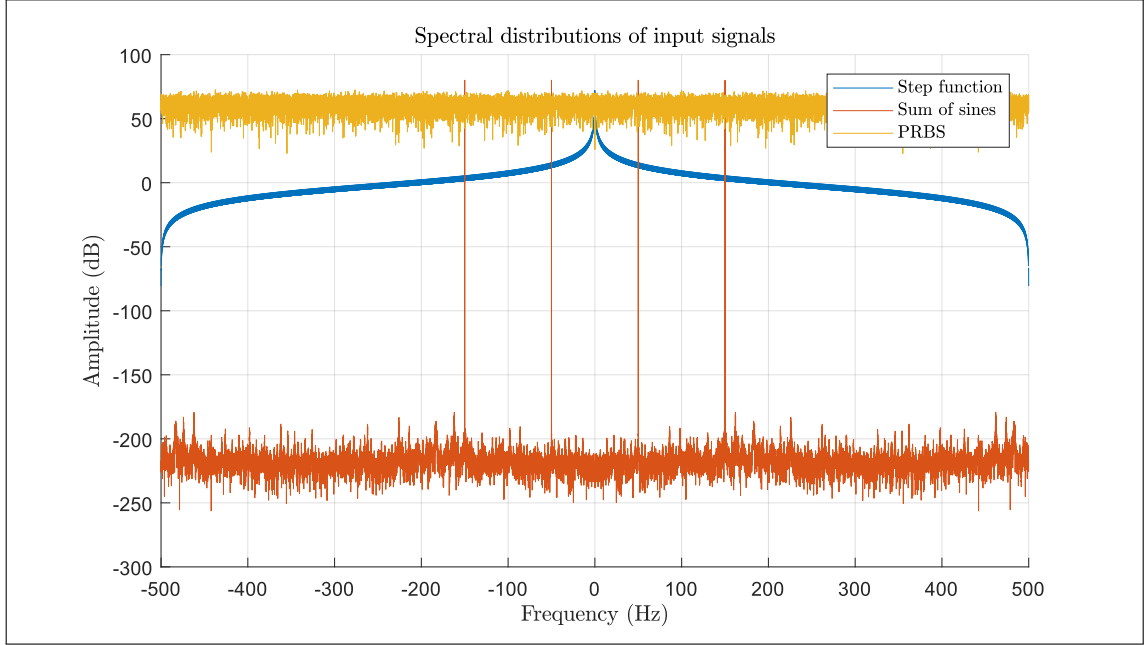


FIGURE 3.2. Spectrum of several input signals

3.7.3. White noise & Pseudo Random Binary Sequences (PRBS)

Given the requirements of the application, the identification input needs to be small enough to not interfere with the PHIL simulation and it needs to have a wide spectral content in order to properly identify the plant. It should also ideally be persistent over time. As such, the perfect identification input for this application is discrete white noise. Discrete white noise is a stationary stochastic process with the characteristic that each sample is not correlated with each other.

Example 3.3. We call **discrete white noise** to a discrete-time random process $X(t)$ if

$$E[X[n]] = 0$$

$$Var[X[n]] = \sigma^2$$

and its autocorrelation function $r_{xx}(\tau)$ only has non-zero value for $\tau = 0$ where $r_{xx}(0) = \delta[0]\sigma^2$

This is particularly useful as it implies that the power spectral density (PSD) of discrete white noise is even across all frequencies, with magnitude σ^2 . This is a direct application of the theorem 3.1. From this it is trivial to prove that the PSD of white noise is a constant σ^2 .

Example 3.4. *Let $e[n]$ be discrete white noise of variance σ^2 . Its autocorrelation function is given by:*

$$r_{ee}(\tau) = \delta(0)\sigma^2$$

Then, taking its discrete-time Fourier transform we obtain.

$$\Phi_{ee}(\omega) = \sum_{\tau=-\infty}^{\infty} R_{ee}(\tau)e^{-i\omega\tau} = \sum_{\tau=-\infty}^{\infty} \delta(0)\sigma^2 e^{-i\omega\tau} = \sigma^2 \cdot \sum_{\tau=-\infty}^{\infty} \delta(0)e^{-i\omega\tau} = \sigma^2 \cdot 1$$

Unfortunately, white noise is notoriously difficult to generate computationally. It is also impossible to reproduce as real white noise cannot exist, given that it has infinity power.

Example 3.5. *The Power can be calculated from its PSD*

$$P = \frac{1}{\pi} \int_{-\infty}^{\infty} \Phi_{xx}(\omega) d\omega \quad (3.41)$$

From this, we can see that given the even PSD of white noise, the integral above would equal ∞

$$P = \frac{1}{\pi} \int_{-\infty}^{\infty} \Phi_{ee}(\omega) d\omega = \frac{1}{\pi} \int_{-\infty}^{\infty} \sigma^2 d\omega = \infty \quad (3.42)$$

In reality phenomena usually associated with white noise, such as thermal noise, correspond to the behaviour of white noise filtered by a low pass filter. For identification purposes, we simply require the input to properly excite all the relevant modes of the

system, meaning that as long as the frequency content remains relatively even in the frequency band of interest, it will provide good identification results, even if the spectrum goes to zero for very high frequencies. Nonetheless, the problem of efficiently generating a white noise analogue still remains. White noise is notoriously difficult to generate using a computer, as true random behaviour is impossible to obtain from a purely deterministic system. Thus, a common analogue for white noise is a pseudo random binary sequence or PRBS.

A PRBS is a binary sequence that seems random for up to N elements until they start repeating themselves, unlike true random sequences, that are infinite. They also have the property that their autocorrelation function has only two values. In fact this property is usually used to define these sequences.

Definition 3.4. *A PRBS is periodic signal of length M that shifts between two values: a and $-a$; and whose autocorrelation function is given by:*

$$r_u(\tau) = \begin{cases} a^2 & \tau = 0, M, 2M, \dots \\ -a^2/M & \text{elsewhere} \end{cases} \quad (3.43)$$

We can see that the autocorrelation function is remarkably similar to that of white noise, provided $a = \sigma$, at least if only one period is considered. This as if the length of the sequence is very large, $-a^2/M$ will be close to zero. Given the insight provided by theorems 3.1 and 3.2, the autocorrelation function and the PSD contain essentially the same information. Meaning that if their autocorrelation functions become more similar as M goes to infinity, so will their respective spectrum. Figure 3.2 shows that indeed, the spectrum of a PRBS is very similar to that of white noise. For further reading about the spectral properties of white noise and PRBS sequences see ([Torsten y Stoica, 1989b](#)).

As mentioned above, PRBS's are useful because they have similar statistical properties to white noise but are easy to generate in a computer. They are generated by circuits called Linear-feedback shift register (LSFR). These circuits are made from a series of

shift registers or delays and XOR gates, figure 3.3 shows the practical implementation of a sequence generator. These circuits can also be represented by mod 2 polynomials. Although there are a large number of possible generating circuits, only a few will produce a sequence with the properties we are looking for. In particular, we want to use circuits that produce maximum length sequences, that is, the go through $2^N - 1$ different values before repeating themselves. Circuits that can be represented by primitive polynomials will produce these types of sequences. The concept of a primitive polynomial is beyond the scope of this thesis and further reading about this topic can be found in (Berlekamp, 1968) and (Weisstein, 2020). Nonetheless, some examples are shown below. These are used to generate a PRBS of a given length.

$$PRBS7 = x^7 + x^6 + 1 \quad (3.44)$$

$$PRBS9 = x^9 + x^5 + 1 \quad (3.45)$$

$$PRBS11 = x^{11} + x^9 + 1 \quad (3.46)$$

$$PRBS23 = x^{23} + x^{18} + 1 \quad (3.47)$$

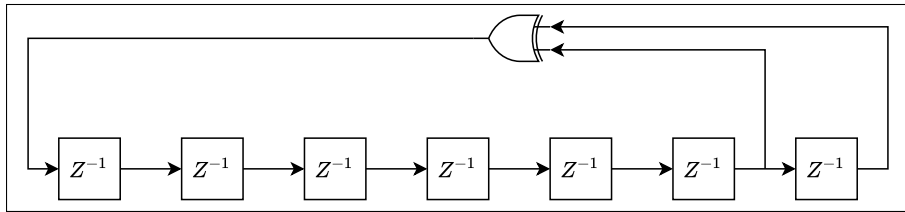


FIGURE 3.3. LFSR implementing PRBS7: $x^7 + x^6 + 1$

3.8. Spectral analysis: implementation and offline results

Given everything stated in previous sections of this chapter, we are ready to implement and test the procedure described in section 3.2. The identification input selected was a

PRBS signal given is statistical similarity to white noise. The identification process can be summarised by algorithm 1.

Algorithm 1: Spectral analysis	
Result: Non parametric model $\hat{G}(e^{i\omega})$ of impedance Z_{HUT}	
Initialization	
Generate PRBS of length 5N	
Apply PRBS;	// Excite the system
Extract measurements: $u(t)$ and $y(t)$	
	// Measure the system
Splits $u(t)$ and $y(t)$ in 5 segments	
Discard 1st segment	
Average 4 remaining segments to get $\bar{u}(t)$ and $\bar{y}(t)$	
Apply equations 3.29 and 3.30 to calculate $\hat{r}_u(\tau)$ and $\hat{r}_{yu}(\tau)$	
	// Correlation analysis
Apply window $w(\tau)$ to $\hat{r}_u(\tau)$ and $\hat{r}_{yu}(\tau)$	
	// Windowing
Use equations 3.40 and 3.39 to calculate $\hat{\Phi}_u$ and $\hat{\Phi}_{yu}$	
Divide $\hat{\Phi}_{yu}$ by $\hat{\Phi}_u$ to obtain $\hat{G}(e^{i\omega})$	// Estimated transfer function

Two scenarios were used to test the identification algorithm. The models used can be seen in figures 3.4 and 3.5 and the parameters can be found in table 3.1 :

1. The first system involves a simple LR filter. To simulate a grid, a 50Hz 220V RMS signal $s(t)$ is given alongside the identification input $id(t)$. The sum of both signals will be $u(t)$. The output of the system, $y(t)$ will be the measured current, thus $\hat{G}(e^{i\omega})$ will be the filter impedance.
2. The second scenario involves an RLC filter. Here, we will identify its transfer function between its input and the voltage of the resistance. Just as in case 1, a 50 Hz 220 VRMS signal is added on top of the identification input.

The N was selected at 10000 for both cases and γ was set at 12.5 % of N . A simulation time of 8 seconds was selected alongside a sample time of $160\mu s$. This allows for $5N$ samples to be collected. The results for cases 1 and 2 are shown in figures 3.6.

TABLE 3.1. Parameters of identified systems

	R	L	C
System 1	$1\ \Omega$	$1mH$	N/A
System 2	$3.3\ \Omega$	$1mH$	$330\mu C$

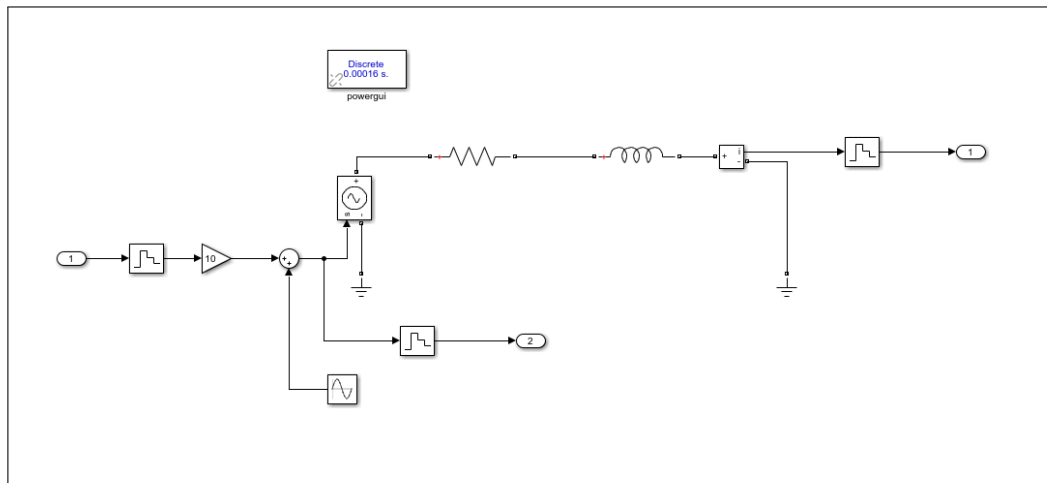


FIGURE 3.4. Identification circuit for case 1: RL

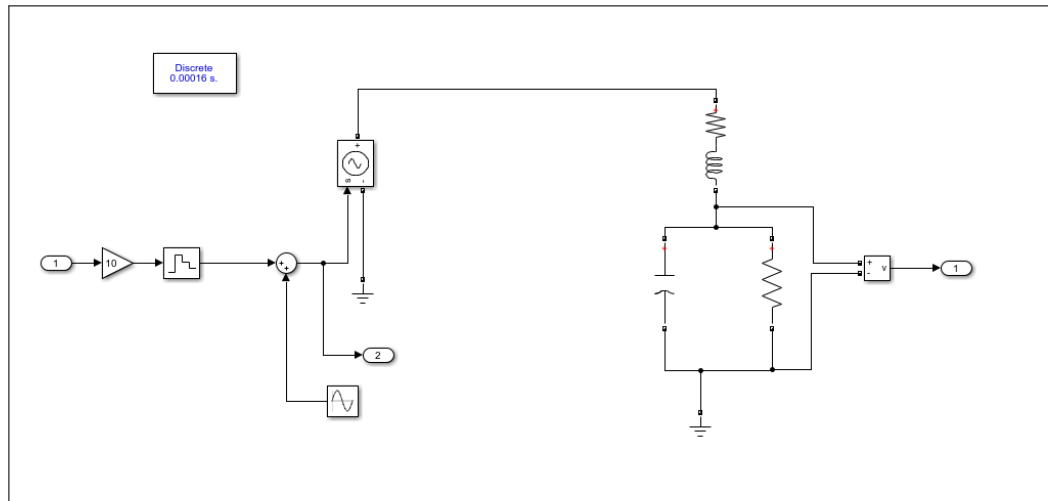


FIGURE 3.5. Identification circuit for case 2: RLC

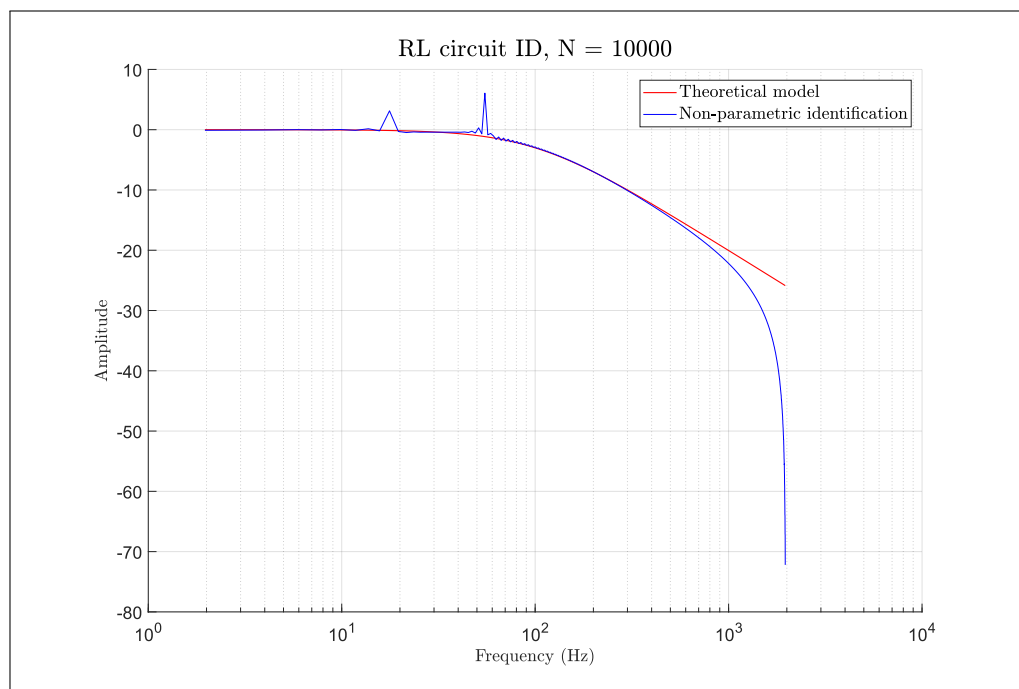


FIGURE 3.6. Results for RL filter identification

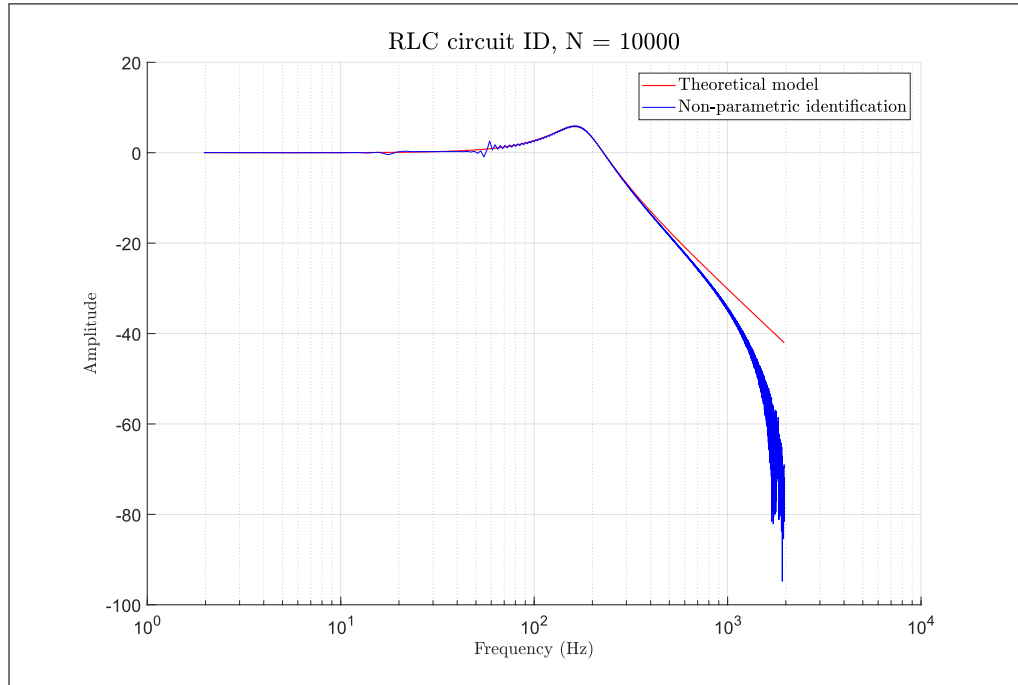


FIGURE 3.7. Results for RLC filter identification

Both test scenarios show the good results as the routine was able to properly identify both circuits. A loss of accuracy can be seen as frequency increases and we approach the Nyquist frequency. This loss of accuracy can be attributed to the windowing function which limits our frequency resolution. Regardless, the results obtained seem appropriate for the frequency band of interest.

Some errors are present in the identification which are to be expected, given that our estimates for the spectrum are random variables that, as N goes to infinity, their mean will converge to the real spectrum and their variance will decrease to zero. This could be a problem if the information obtained from our spectral analysis was used directly, this will not be the case as this information will be used to obtain a parametric identification of our system.

3.9. Parametric identification

In the previous section we have obtained a non parametric model of our impedance, however a problems remains. Our true goal is to identify the impedance of the HUT in a PHIL simulation and to implement that impedance into the DIM interface algorithm. For this, we need a model of Z_{HUT} that is simple to implement. Thankfully, the information obtained through our spectral analysis allows for a very straight forward approach. If we wish to obtain the parameter of the transfer function of impedance, we must only fit a rational function to the date generated by the spectral analysis.

To do this, we can solve a least squares problem to make the data fit a rational function. To use least squares for this we must adapt the problem as seen in [Bañuelos, Gutiérrez, y Gustavsen \(2017a, pp. 7–17\)](#). We start by defining our model structure in $\frac{N(s)}{D(s)}$.

$$F(s) \cong \frac{N(s)}{D(s)} = \frac{a_0 + a_1s + a_2s^2 + \dots + a_ns^n}{1 + b_1s + b_2s^2 + \dots + b_ns^n} \quad (3.48)$$

As we want to fit $\frac{N(s)}{D(s)}$ to the data $F(s)$, we can define an error term as follows.

$$\epsilon = F(s) - \frac{N(s)}{D(s)} \quad (3.49)$$

Multiplying both sides by $D(s)$ we get:

$$\epsilon' = \epsilon D(s) = F(s)D(s) - N(s) \quad (3.50)$$

Then equating everything to zero.

$$F(s)(1 + b_1s + b_2s^2 + \dots + b_ns^n) - (a_0 + a_1s + a_2s^2 + \dots + a_ns^n) = 0 \quad (3.51)$$

From here, our unknown coefficients can be found. Equation 3.51 gives us one equation for every data point k . This implies we will have an overdetermined system with k equations of the form $Ax = b$ where:

$$A_k = [1 \quad s_k \quad \cdots \quad s_k^n \quad -s_k F(s_k) \quad \cdots \quad -s_k^n F(s_k)] \quad (3.52)$$

$$x = [a_0 \quad a_1 \quad \cdots \quad a_n \quad b_1 \quad b_2 \quad \cdots \quad b_n]^T \quad (3.53)$$

$$b = [F(s_1) \quad \cdots \quad F(s_k)]^T \quad (3.54)$$

We know that s and $F(s)$ are complex, meaning that A and b will be complex as well. To apply the least squares methodology we must obtain purely real matrices. To do this we will evaluate A and b and then split the real and imaginary parts.

$$A_r = \text{real}(A) \quad (3.55)$$

$$A_i = \text{imag}(A) \quad (3.56)$$

$$b_r = \text{real}(b) \quad (3.57)$$

$$b_i = \text{imag}(b) \quad (3.58)$$

From this, we form new matrices A_n and b_n .

$$A_n = \begin{bmatrix} A_r \\ A_i \end{bmatrix} \quad (3.59)$$

$$b_n = \begin{bmatrix} b_r \\ b_i \end{bmatrix} \quad (3.60)$$

Now our system $A_n x = b$ is purely real and we can apply the least squares method. The objective function of the least squares problem is, as always:

$$\min_x ||A_n x - b_n||_2 \quad (3.61)$$

Where the solution for x is given by

$$A_n^T A_n x = A_n^T b_n \quad (3.62)$$

One last useful technique for solving equation 3.62 is to use QR decomposition. If A is an $m \times n$ matrix then, $A = QR$, where Q being $m \times n$ and R is a upper triangular matrix of $n \times n$. Also $Q^T Q = I$.

With this 3.62 becomes :

$$[QR]^T QR x = [QR]^T b_n \quad (3.63)$$

$$R^T Q^T Q R x = R^T Q^T b_n \quad (3.64)$$

$$R^T R x = R^T Q^T b_n \quad (3.65)$$

$$R x = Q^T b_n \quad (3.66)$$

With this final equation 3.66 we can efficiently solve for x and get the parameters we need for our model. Results for the parametric identification of cases 1 and 2 can be seen in figures 3.8 and 3.9. Initially this seemed to work as the results of the identification were good when $N = 10000$. The spectral analysis produced good results and in turn the least squares method was able to produce accurate parameters.

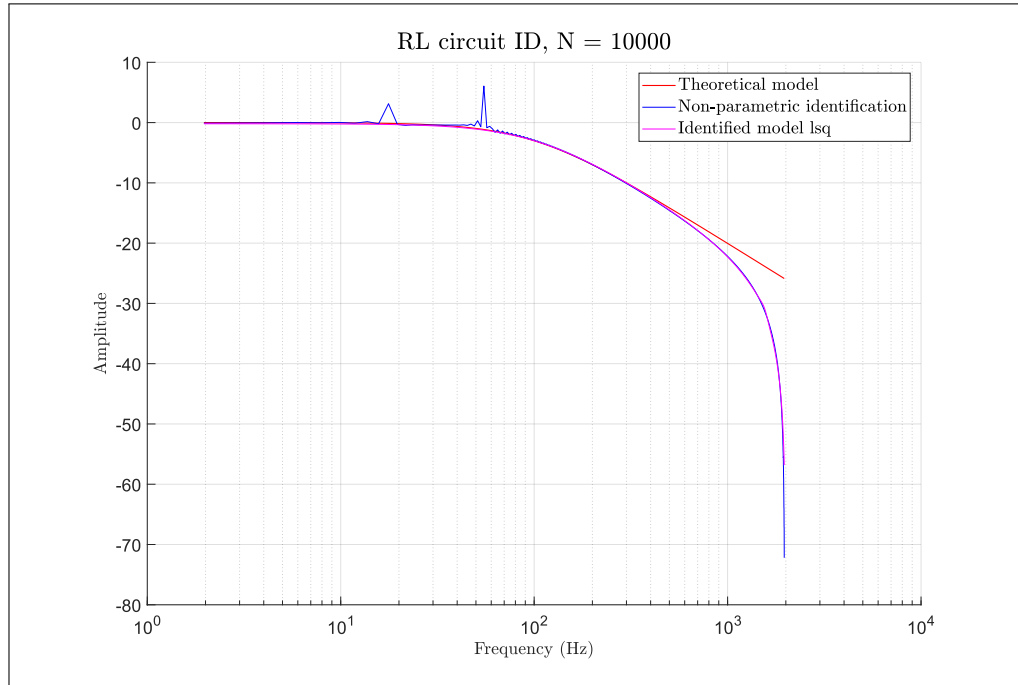


FIGURE 3.8. Results for RL filter identification

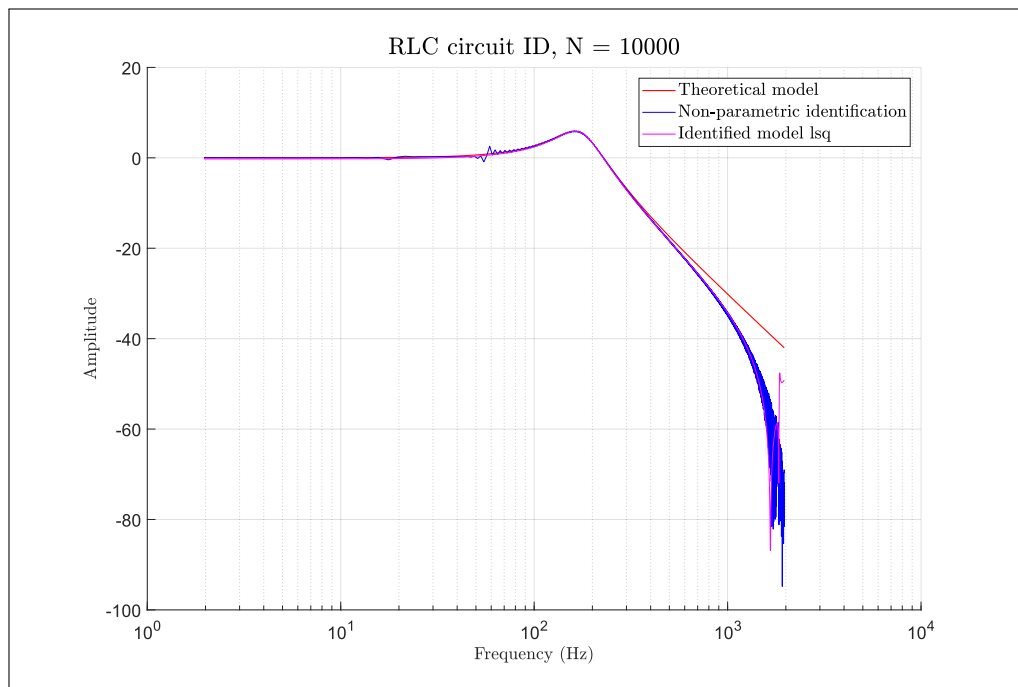


FIGURE 3.9. Results for RLC filter identification

Nonetheless, this changes as N is reduced. This causes the quality of the spectral analysis data to deteriorate and heavily impacts the performance of the least squares method. We can see this in figure 3.10.

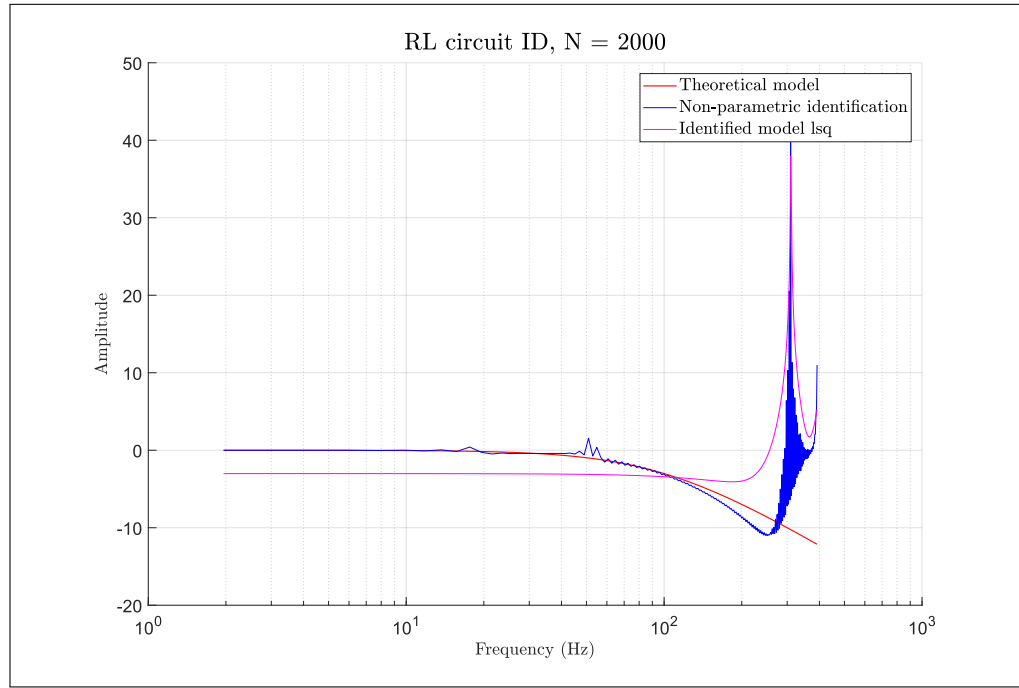


FIGURE 3.10. Least square ID. $N = 2000$

This is to be expected as literature says the least squares method is generally biased towards higher frequencies and given that these can be poorly estimated by the spectral analysis, using the least squares technique may prove unreliable. Thus, we require a new method for solving the parameter fitting problem.

3.10. Levenberg-Marquardt method

The Levenberg-marquardt (LM) method is an optimisation based technique that iteratively improves the parameters of a function to minimise the sum of the square error between it and a given data set. This method was proposed by Levenberg ([Levenberg, 1944](#)) and Marquardt ([Marquardt, 1963](#)) as a combination of two of the most popular

minimisation methods: the Gradient Descent (GD) method and the Gauss-Newton (GN) method.

Consider the model $\hat{F}(s, x)$ as a function of frequency s and a set of parameters $x = [x_1, x_2, \dots, x_n]$ which needs to be fitted to the data $F(s)$. The objective function of our optimisation problem will be given by:

$$\gamma(x) = \frac{1}{2} \sum_{k=1}^{N_s} (\hat{F}(s_k, x) - F(s_k))^2 \quad (3.67)$$

Where N_s is the number of samples in our data set. Further manipulation of equation 3.67 gives:

$$\gamma(x) = \frac{1}{2} (\hat{F}(s_k, x) - F(s_k))^T (\hat{F}(s_k, x) - F(s_k)) \quad (3.68)$$

$$\gamma(x) = \frac{1}{2} \hat{F}(s_k, x)^T \hat{F}(s_k, x) - \hat{F}(s_k, x)^T F(s_k) + \frac{1}{2} F(s_k)^T F(s_k) \quad (3.69)$$

As with all these methods, the goal is to find in each iteration a perturbation h for the parameters x that decreases the value of $\gamma(x)$. The LM method does this by combining the updates formulas of the of the GN and GD methods.

The gradient descent method, as its name implies, calculates h_{GD} so that the change in parameters produces the steepest descent. Thus the update equation for this method id given by:

$$h_{GD} = -J^T (\hat{F}(s_k, x) - F(s_k)) \quad (3.70)$$

This method behaves well when we are far from the optimal solution, approaching quickly to the optimal solution. Nonetheless, when close to it convergence slows down significantly as the gradient flattens.

The Gauss-Newton method, on the other hand assumes the $\gamma(x)$ is approximately quadratic near the solution, hence it uses a first order Taylor series to approximate the model.

$$\hat{F}(s, x + h) \approx \hat{F}(s, x) + \frac{\partial \hat{F}(s, x)}{\partial x} h = \hat{F}(s, x + h) + Jh \quad (3.71)$$

We can substitute equation 3.71 in equation 3.68 and obtain:

$$\gamma(x + h) = \frac{1}{2} \hat{F}^T \hat{F} - \hat{F}^T F + \frac{1}{2} F^T F + (\hat{F} - F)^T Jh + \frac{1}{2} h^T J^T Jh \quad (3.72)$$

Taking the derivative with respect to h and equating to 0 to minimise $\gamma(x)$, we get.

$$0 = (\hat{F} - F)^T J + h^T J^T J \quad (3.73)$$

$$h_{GN} = (J^T J)^{-1} J^T (\hat{F} - F) \quad (3.74)$$

This approach behaves very well when close to the solution but it will be very slow to converge if we start far from it as it will not take large steps as the GD method.

The Levenberg Marquardt method combines the strengths of both the previous methods, behaving as the GD method when far from the local minimum and like the GN method when close to it, achieving overall faster convergence. This is done by adaptatively changing a parameter λ call the Levenberg-Marquardt parameter. The update equation for the LM method can be seen in 3.75.

$$(J^T J + \lambda I) h_{LM} = -J^T (\hat{F} - F) \quad (3.75)$$

From the update equation it is clear that when λ is large, the method will behave like the GD method, as the contribution of the hessian will be overpowered by λI . Then as

λ decreases the method will resemble more and more the GN method. A modification of equation 3.75 where I is replaced by the diagonal of the hessian matrix is also very common in the literature.

$$(J^T J + \lambda \text{diag}(J^T J))h_{LM} = -J^T(\hat{F} - F) \quad (3.76)$$

The only problems that remains then, is how to update λ to achieve the behaviour described above. While different implementations of the LM method have different method to update they all follow the same strategy: when $\gamma(x)$ decreases so does λ . Following this general rule, the update of λ will be performed as follows:

- If for any given iteration k $\gamma(x_k + h_k) < \gamma(x_k)$ then the changes to x are accepted and $x_{k+1} = x_k + h_k$. We also decrease λ by: $\lambda_{k+1} = \lambda_k * \alpha$ where $\alpha < 1$.
- On the contrary if $\gamma(x_k + h_k) > \gamma(x_k)$ then the changes to the parameters are not accepted and $x_{k+1} = x_k$. λ is then increased by: $\lambda_{k+1} = \lambda_k * \beta$ where $\beta > 1$.

Finally, we must take into account the fact that γ will be complex functions, and so will J as $F(s)$ is complex. To overcome this we will apply the same technique as in the regular least squares method and take their real and imaginary parts to form a new vector $\gamma_n(s)$ and Jacobian J_n . With this, the process to apply the LM method can be summarised in algorithm 2.

Result: Parametric model of impedance Z_{HUT}

```
while Stop conditions are not met do
```

```
// Calculate error function
```

```
// Calculate Jacobian
```

```
// Form  $\gamma_n$  and  $J_n$ 
```

```
// Calculate perturbation
```

$$x_{new} = x + h$$
$$\lambda = \lambda * \alpha$$
$$\lambda = \lambda * \beta$$

```
exit inner loop
```

```
// Update  $x$  and  $\lambda$ 
```

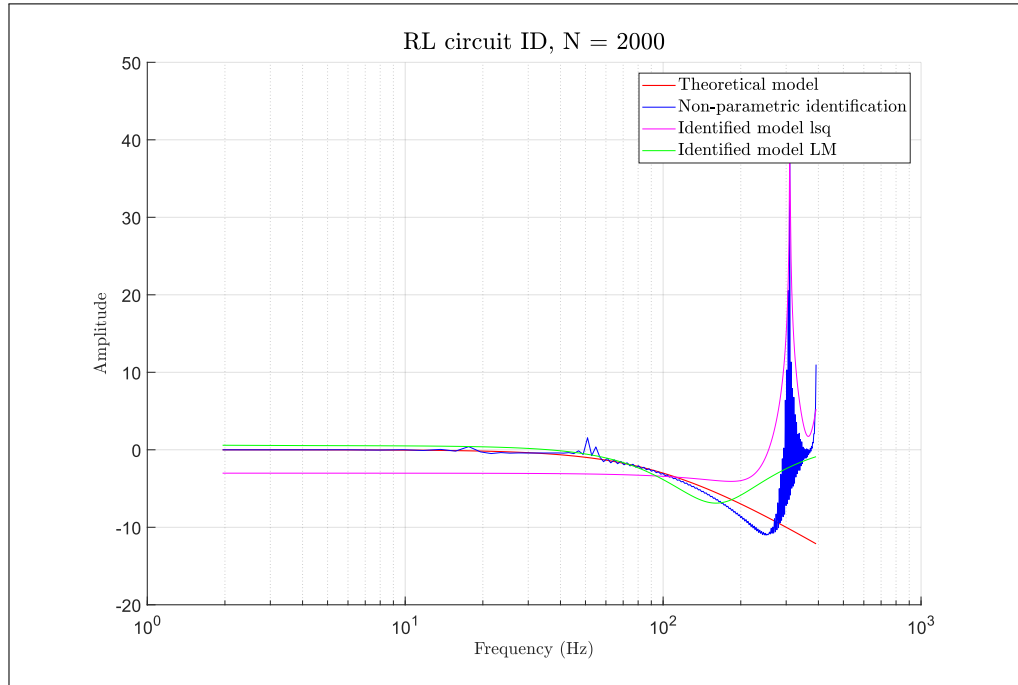


FIGURE 3.11. RL circuit ID with LM method. $N = 2000$

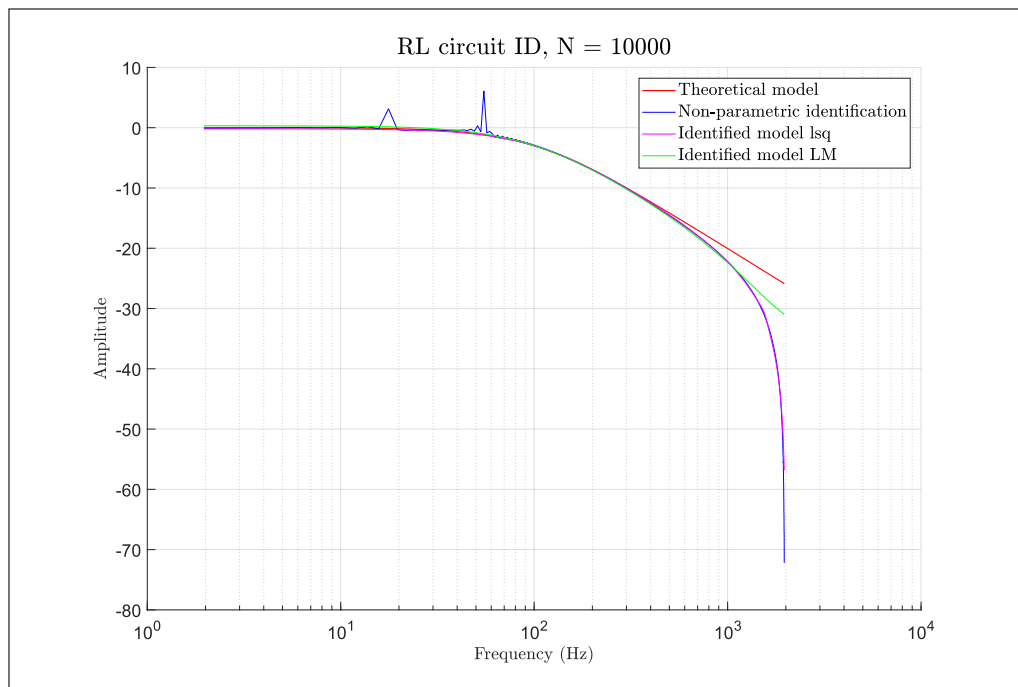


FIGURE 3.12. RL circuit ID with LM method. $N = 10000$

3.11. Chapter conclusion

Throughout this Chapter we have described the the mathematical tools needed for the implementation of the proposed identification routine as well as the sequence of actions needed to obtain useful model for the HUT's impedance. In particular we have seen how windowing is crucial in order to obtain good results out of spectral analysis as the estimations given often have high variance. To deal with this problem and size of the window γ , as well as the number of samples N of the recording must be selected with care and be subject to validation and testing. The trade off between frequency resolution and reduction of variance implicit in the choice of window size must be kept in mind. Once the non-parametric identification is done, obtaining a good model out of it requires a robust parameter fitting routine. While regular least-squares approach appeared to provide good results when provided clean data by the previous identification stage, as soon as quality decreased it stopped providing useful results. This method was also seen to be biased towards fitting the higher frequency components as more data point are placed there. By contrast the LM method proved much more reliable, providing useful results even in the presence of noisy or poor quality data from spectral analysis. Thus, the LM method was selected to for the parameter fitting stage of the identification routine, despite its higher computational cost.

4. SOFTWARE IMPLEMENTATION & SIMULATION RESULTS

4.1. Introduction

Chapters 2 and 3 laid out the different concepts and tools needed to implement a PHIL platform. The damping impedance method was selected as our interface algorithm of choice and a process to overcome its mayor disadvantage, the fact that we need to identify Z_{HUT} , was presented and tested. This Chapter will cover the implementation and testing of PHIL platform in a Matlab Simulink simulation while Chapter 5 will show the hardware implementation of the real PHIL platform using the model developed in this chapter as a base.

A PHIL platform has 3 mayor elements, a real simulation target, a power amplifier and a HUT. Figure 4.1 shows the structure of the implemented PHIL platform.

The real time simulator can be seen in blue. It runs all the electrical simulations, the identification algorithm, handles the I/O and processes the measurements for immediate use or for recording. Normally a real time target would have trouble handling many different tasks as the electrical simulation requires a very short time step, leaving little processing time for other functions. In our implementation however, we take advantage of the many processing cores of the real time target, to run each process independently. This simplifies the implementation as the job of multiple devices is done by a single one.

The amplifier, seen in figure 4.1 in red, is the link between the real part of the simulation, in this case, the HUT, and its software side. Its job is to take the voltage commands coming from the simulation and turn them into high voltage and high power signals, to apply them to the HUT. This device must be very fast at responding to voltage commands, as significant delays in reaching the desired set point will heavily compromise the stability of any interface algorithm. It must also be powerful enough to supply or sink the power the HUT demands.

Finally, the HUT is the device or circuit with which we want to interface our simulation and it is directly connected to the amplifier. Its voltage and current must be measured at the connection point so they can be fed back into the simulation.

All these components will be simulated to test the performance of the identification routine and the interface algorithm.

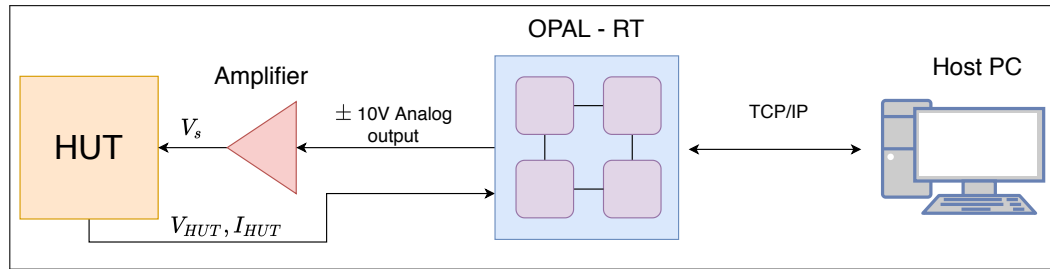


FIGURE 4.1. PHIL platform structure

4.2. Opal-RT model structure

The tool we will use to build the software for the real time target, RT-LAB, requires us structure our model in an specific way. Given that the Simulink model built in this chapter will be the basis of the hardware implementation, we will use the same structure . The program requires us to split the model in subsystems. It then assigns each subsystem to a different core. The tool allows for 3 types of subsystem which must be differentiated by their name as seen in figure 4.2. Every model must have one master subsystem denoted by SM before its name. Then systems may have multiple slave subsystems, denoted by the SS before their name. Both master and slaves subsystems will be compiled into an executable file to run in real time and will be assigned each to a different core if available. The communication between these subsystems is synchronous, meaning that during the real time execution the timing of each core will be consistent with all the other. Each subsystem may have its own different timestep, however all timesteps in a given model have to be a multiple of the smallest one.

The system may also have one console subsystem, denoted by SC before its name, this subsystem will not be compiled and will not be loaded on to the real time target. Instead it will run in the host PC and it is used to display measurements and send user commands of the simulation. Unlike the communication between SM and SS subsystems, the communication between the host and the real time simulation is asynchronous, meaning accuracy in timing is not guaranteed. Because of this the console subsystem may only display information sent from the real time simulation or have controls that allow the user to interact with it. No calculations or processing is allowed in this subsystem. Further reading about the structure needed for real time compatible models and other related topics can be found in OPAL-RT's extensive *Resource Center* ([OPAL-RT, 2020b](#)).

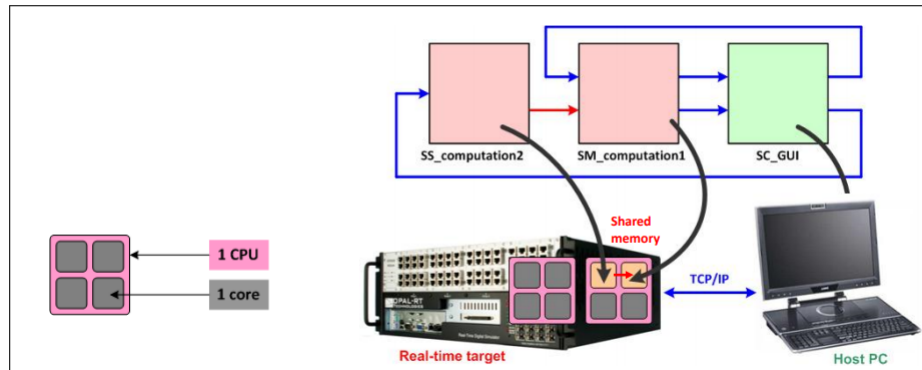


FIGURE 4.2. OPAL RT model structure

This will mean that the top view of our model will correspond to 4 subsystems: 3 for computations and 1 for monitoring. The structure can be seen in figure 4.3 and the implementation in figure 4.4. SM_Computation is the subsystem where the electrical simulation occurs, SS_ID runs the identification routine and SS_RMS calculates power and RMS values. The rest of this chapters explores the implementation of each of these subsystems that will be assigned to each core in the of the real time target.

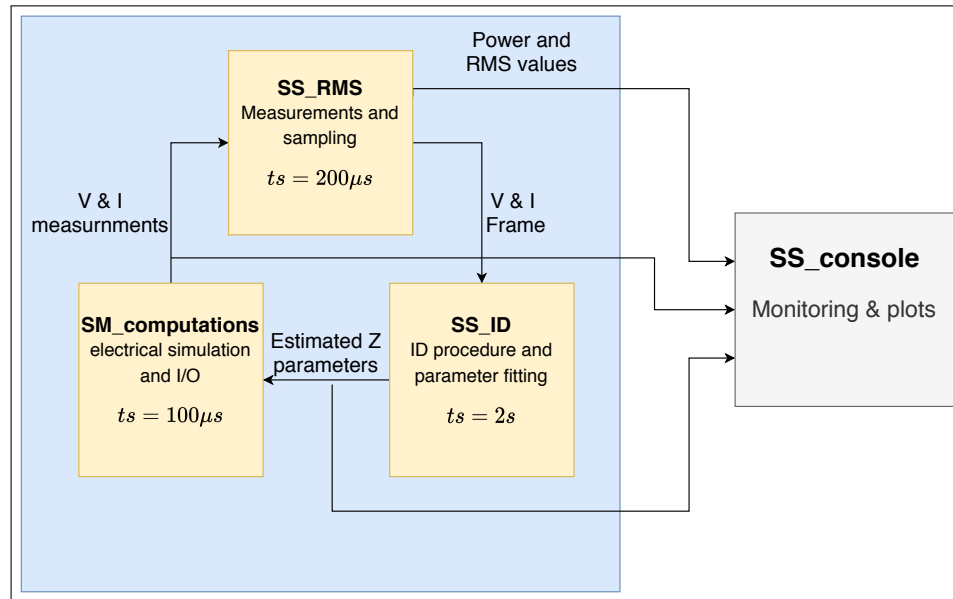


FIGURE 4.3. Software structure

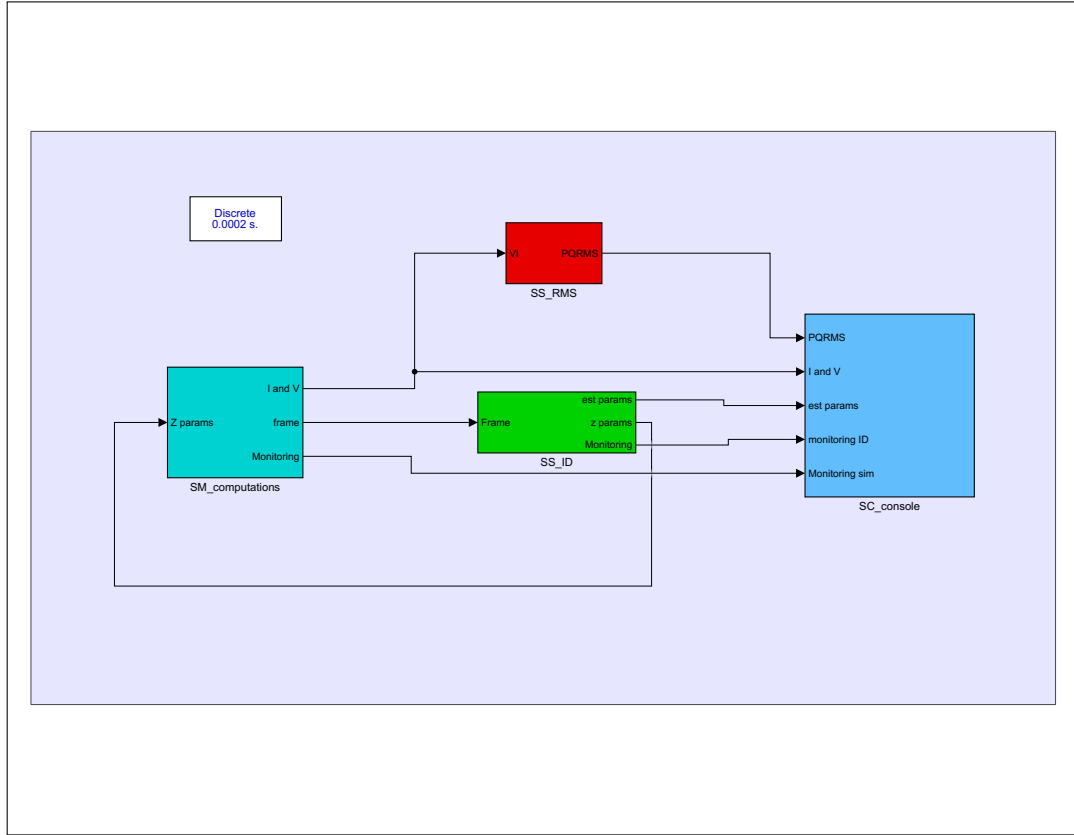


FIGURE 4.4. Top view of Simulink model

4.3. Software implementation: Electrical simulation

The subsystem `SM_computations`, seen in cyan in figure 4.4, will be dedicated to running the electrical simulation. This subsystem will contain the entire simulated network that we wish to interface with the HUT. It will also contain the implementation of our interface algorithm meaning that in the hardware implementation, the analog outputs controlling the voltage amplifier and the analog inputs coming from the voltage and current sensors. Figure 4.5 shows an overview of the subsystem. In it, we can clearly distinguish three parts:

1. Our implementation of the DIM interface algorithm highlighted in violet and seen in figure 4.6.

2. The amplifier and HUT, highlighted in red and purple and seen in figure 4.8
3. A data sampler highlighted in green and seen in figure 4.9.

In figure 4.6 we can see all the parts of the DIM algorithm described in section 2.5. The subsystem shown in green, labelled ROS or rest of system, contains the simulated network to which we want to connect our HUT. Thus making the variables I_{sim} and V_{DIM} the current and voltage of the point of connection. We also see Z_{ab} representing the impedance of the amplifier in the software side. An extra high value resistor is added in parallel to avoid simulation problems caused by having an inductor and a current source in series. Next we see both controlled sources being controlled by measurements coming from the HUT.

Finally Z_{DIM} is implemented using a subsystem. The impedance Z_{DIM} changes during the simulation, to match identified value of Z_{HUT} . Given that the HUT could have any unknown topology, implementing Z_{DIM} with electrical components is impossible. Instead Z_{DIM} is implemented as shown in figure 4.7. First, the voltage between the terminals of the subsystem is measured, this voltage filtered to remove any high frequency noise and is then passed through a time varying discrete transfer function block. This block receives its parameters from the identification subsystem and implements the identified dynamic. A delay block is placed before the transfer function block to prevent algebraic loops, as these cannot exist in real time simulations. This block allows us to mimic the dynamic of any HUT, regardless of its topology.

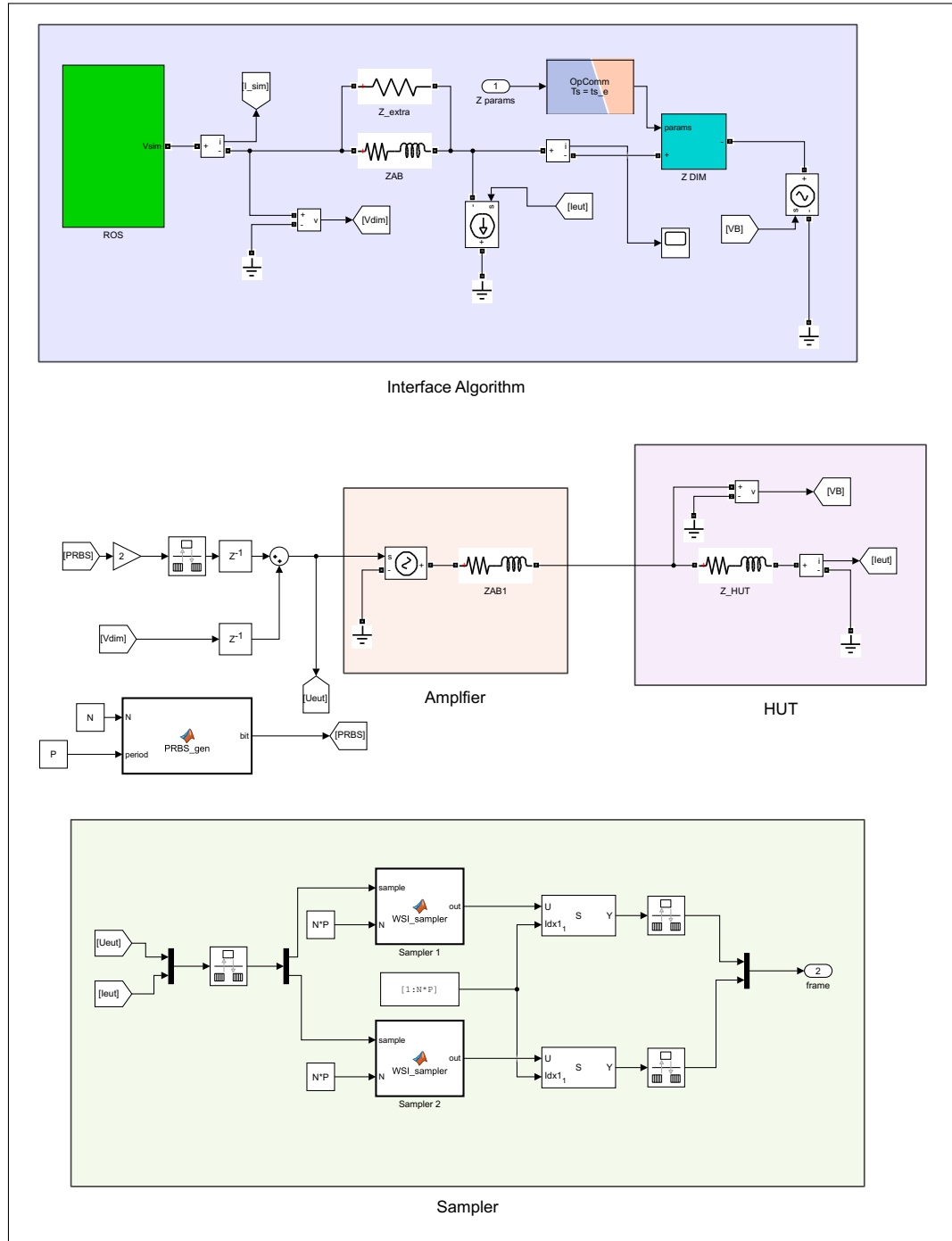


FIGURE 4.5. Overview of subsystem SM computations

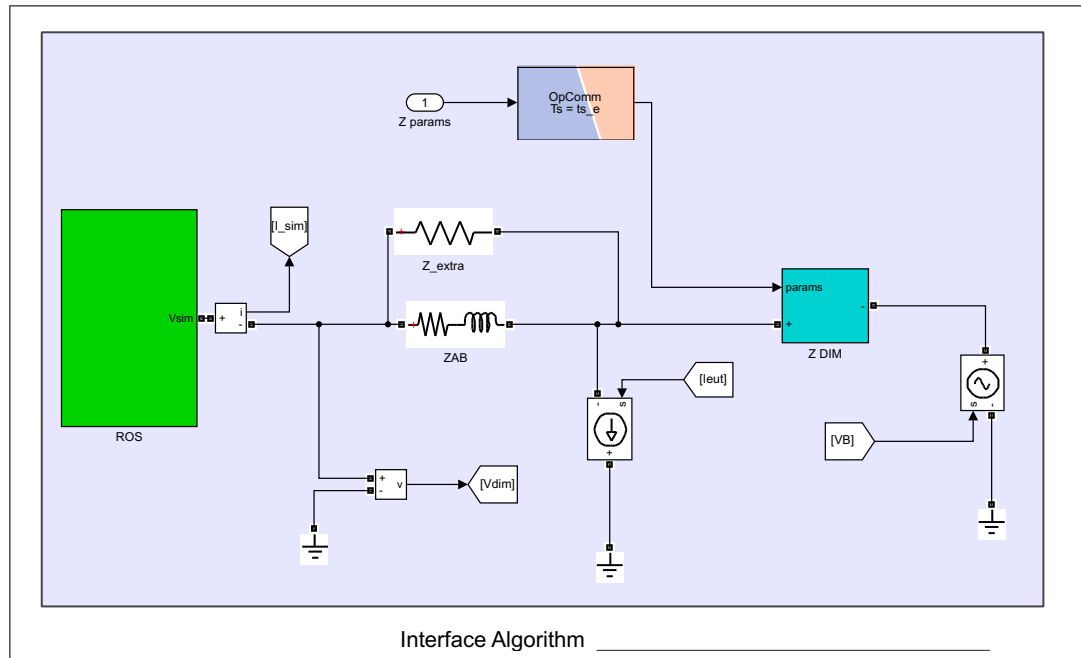


FIGURE 4.6. Implementation of the DIM algorithm

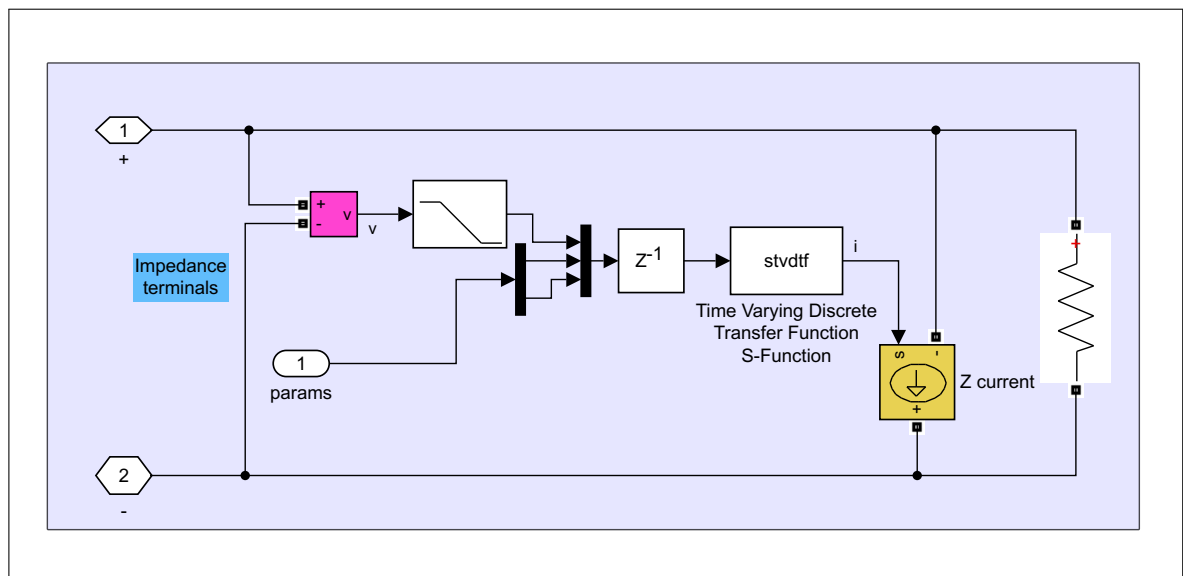


FIGURE 4.7. Implementation of a variable impedance

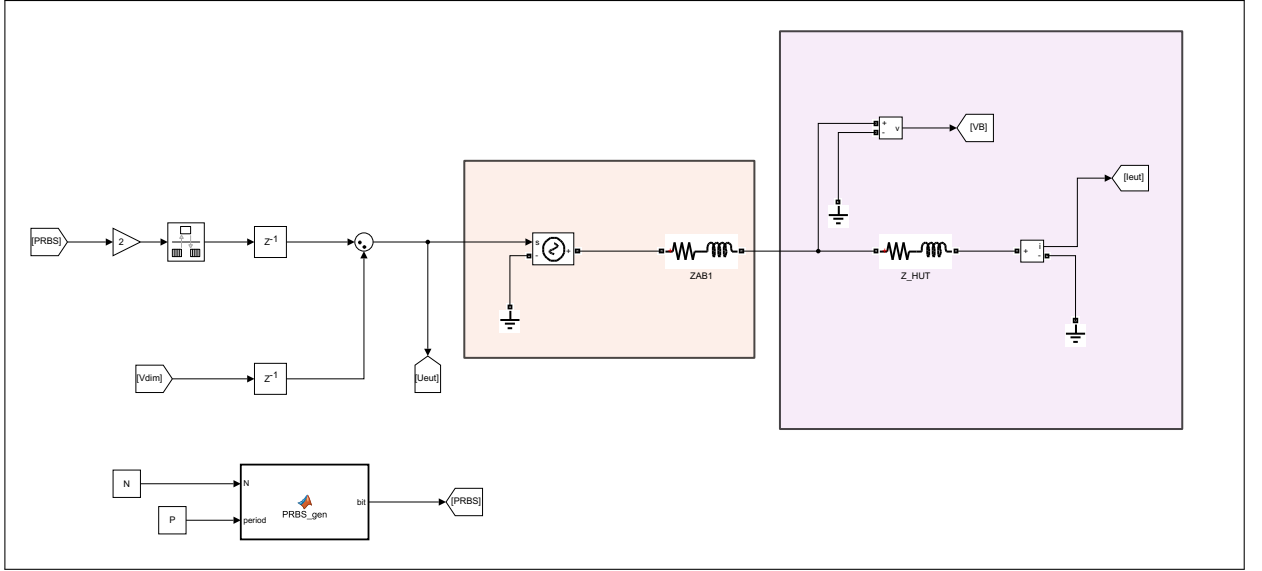


FIGURE 4.8. Amplifier and HUT

As shown in section 2.5, V_{DIM} is perhaps the most important signal as it is the output of the simulation as well as the input to the amplifier, shown in red. To allow the system to identify the HUT a PRBS is added to V_{DIM} before being fed to the amplifier. The amplifier is simply modelled as a controlled voltage source with an input delay. The HUT, shown in red, is an RL filter in this case. In reality the HUT could be any number of complex devices, but an RL filter allows for a direct comparison of the results with the ones obtained in the offline identification.

The last part of this subsystem is the voltage and current sampler. Given that our identification routine needs a large chunk of data at once to perform the identification routine, this part of the subsystem samples the voltage and current of the HUT and records it until it has collected $N \cdot P$ samples, where P is the number of sections that will be averaged, and N is the length of each section. Once it has finished, it updates its output presenting the $N \cdot P$ samples to the identification routine all at once.

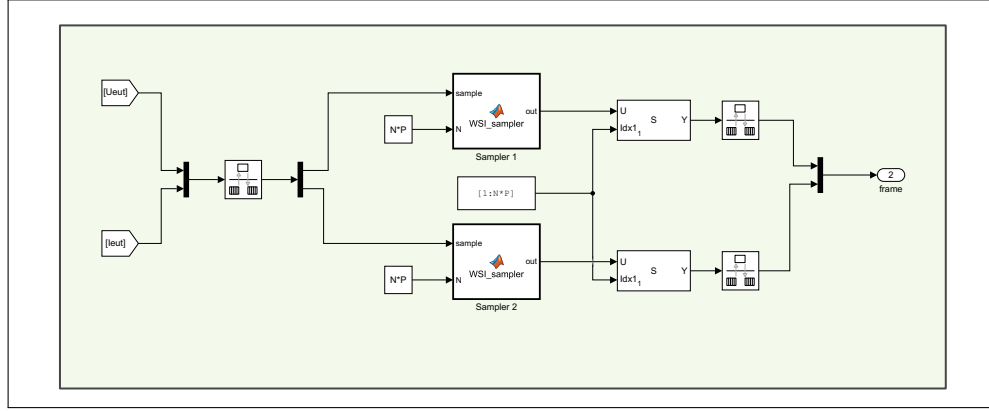


FIGURE 4.9. Implementation of data sampler

4.4. Software implementation: Identification Routine

This subsystem `SS_ID`, seen in figure 4.10, implements the identification process described in chapter 3. It uses the data frame received from the electrical simulation subsystems with the samples of V and I and uses them to perform spectral analysis and then obtain parameters for Z_{DIM} .

Both the spectral analysis routine and the parameter fitting routine are implemented using interpreted functions that execute a Matlab function in an `m` file. The spectral analysis routine receives V and I and uses them to calculate $G(e^{i\omega})$ as a complex curve. Then it passes $G(e^{i\omega})$ and the complex frequency vector to the parameter fitting routine. This routine fits rational functions of several different orders, up to 4^{th} order, to the data of $G(e^{i\omega})$ using the LM method, choosing the best fit to obtain parameters for Z_{DIM} . To do this, the LM method requires an initial guess for the parameters each time we wish to identify the HUT. Naturally on the first identification this guess will be given by the user, however on each subsequent identification, the previous identified values will be used. This is helpful as even if the initial guess is poor due to lack of knowledge of the HUT and first identification produces a poor model, the next time the routine is ran, it will have a much better starting point.

Since the parameters identified through this process correspond to that of a continuous time transfer function, and we implemented Z_{DIM} with a discrete model, the identified transfer function needs to be discretized. The final block of the subsystem performs this discretization by applying the tustin method ($s \approx \frac{2 \cdot (z-1)}{T \cdot (z+1)}$). Finally the system outputs the parameters our variable impedance implementation requires. The basis for the implementation of the LM method in Matlab can be found in (Bañuelos, Gutiérrez, y Gustavsen, 2017b, pp. 17–44)

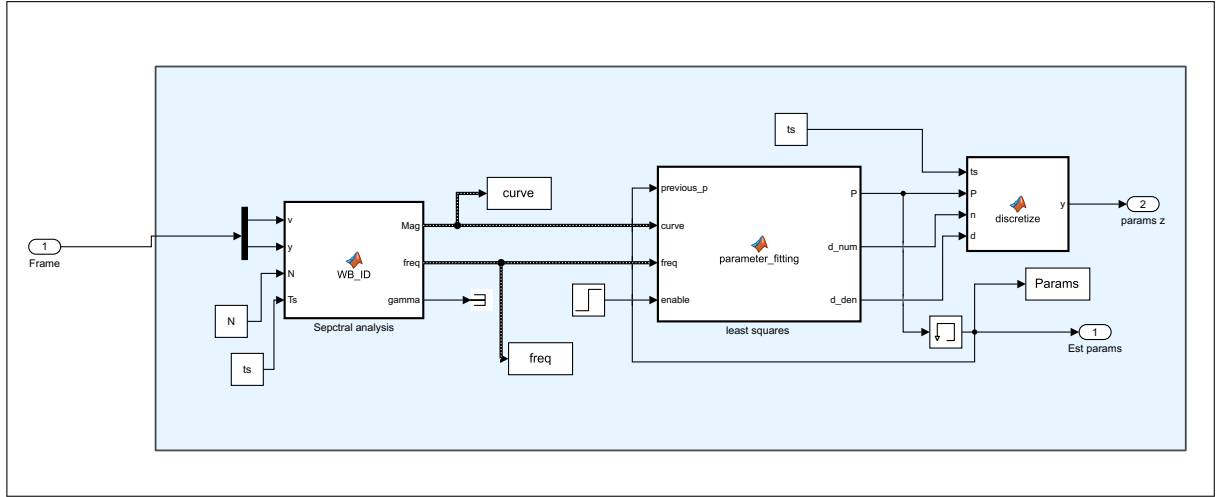


FIGURE 4.10. Implementation of the ID routine

4.5. Software implementation: Measurements

The final subsystem of the model is the measurements subsystem seen in figure 4.11. This subsystem takes the measurements from the electrical simulation and calculated electrical quantities of interest such as active and reactive power as well as RMS values of voltages and currents.

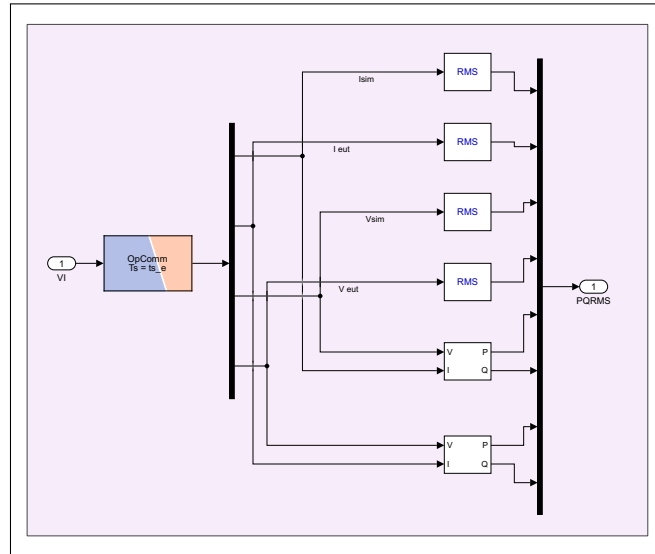


FIGURE 4.11. Implementation of the DIM algorithm

4.6. Simulation results

The model described in the previous sections, while meant to run in Simulink, is designed to exhibit the same behaviour as the final model to be deployed in the real time target. The physical parts of the simulation such as the HUT and amplifier were modelled to resemble their real counterparts.

To consider a PHIL simulation successful, it must achieve the two goals described in chapter 2: be both stable and accurate. In section 2.5 we established that when using the DIM algorithm both stability and accuracy are tied to a proper identification of Z_{HUT} . Given that in the simulation environment we know the HUT we can establish a baseline and evaluate the quality of the identification. This baseline is obtained by simulating the circuit with out the PHIL interface, ie, the naturally coupled system (NCS). This will give us reference values for voltage, current and power to assess the quality of the simulation. To call the simulation a success, it must adequately identify the impedance of the HUT, and provide an electrical simulation with results within a few % point of error with respect to the naturally coupled system. To test the simulation, two scenarios where constructed: one with a passive load and one with an active load. The results are shown below.

4.6.1. Passive load

The tested scenario uses as HUT and LR filter where $R = 8,8\Omega$ and $L = 9mH$, so it can be later compared to a real circuit. The ROS was a 220V RMS voltage source with a 5Ω impedance in series. A simplified diagram of the circuit is presented in figure 4.12. Further relevant parameters can be found in table 4.1.

We can see that that the HUT is properly identified by second 6, as the software side and hardware side variables all equalise. This can be confirmed if we plot the identified parameters and compare it with the theoretical bode. Figure 4.13 shows the identification process at work. We see that the initial guess given was very bad. This resulted the first identification iteration to provide a poor model for Z_{HUT} . However, the next iteration uses

previous parameters as its starting point, leading to a successful identification in the second attempt. Following identifications either improve the model performance or remain within close proximity as seen in figure 4.14.

TABLE 4.1. Simulation parameters: passive load

Parameter	Value
N	2000
P	5
Window size (γ)	250
SM_computations timestep	$100\mu s$
SS_RMS timestep	$200\mu s$
SS_ID timestep	$2s$
Software impedance (Z_s)	5Ω

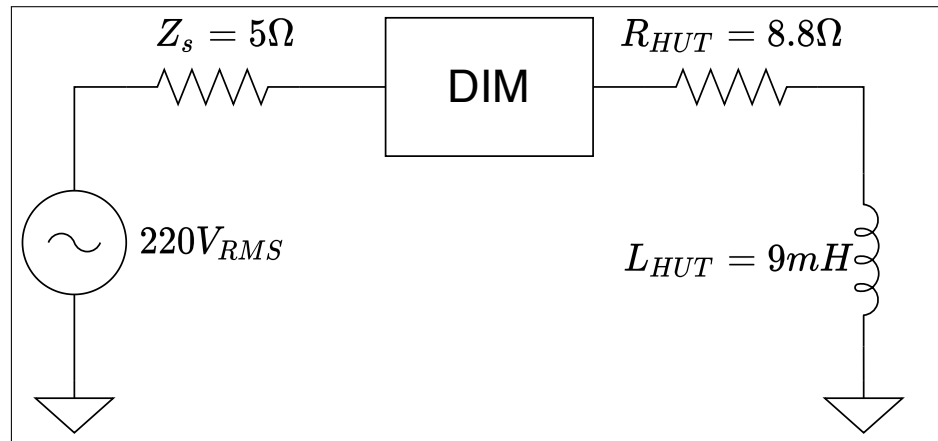


FIGURE 4.12. Simplified diagram of the simulation

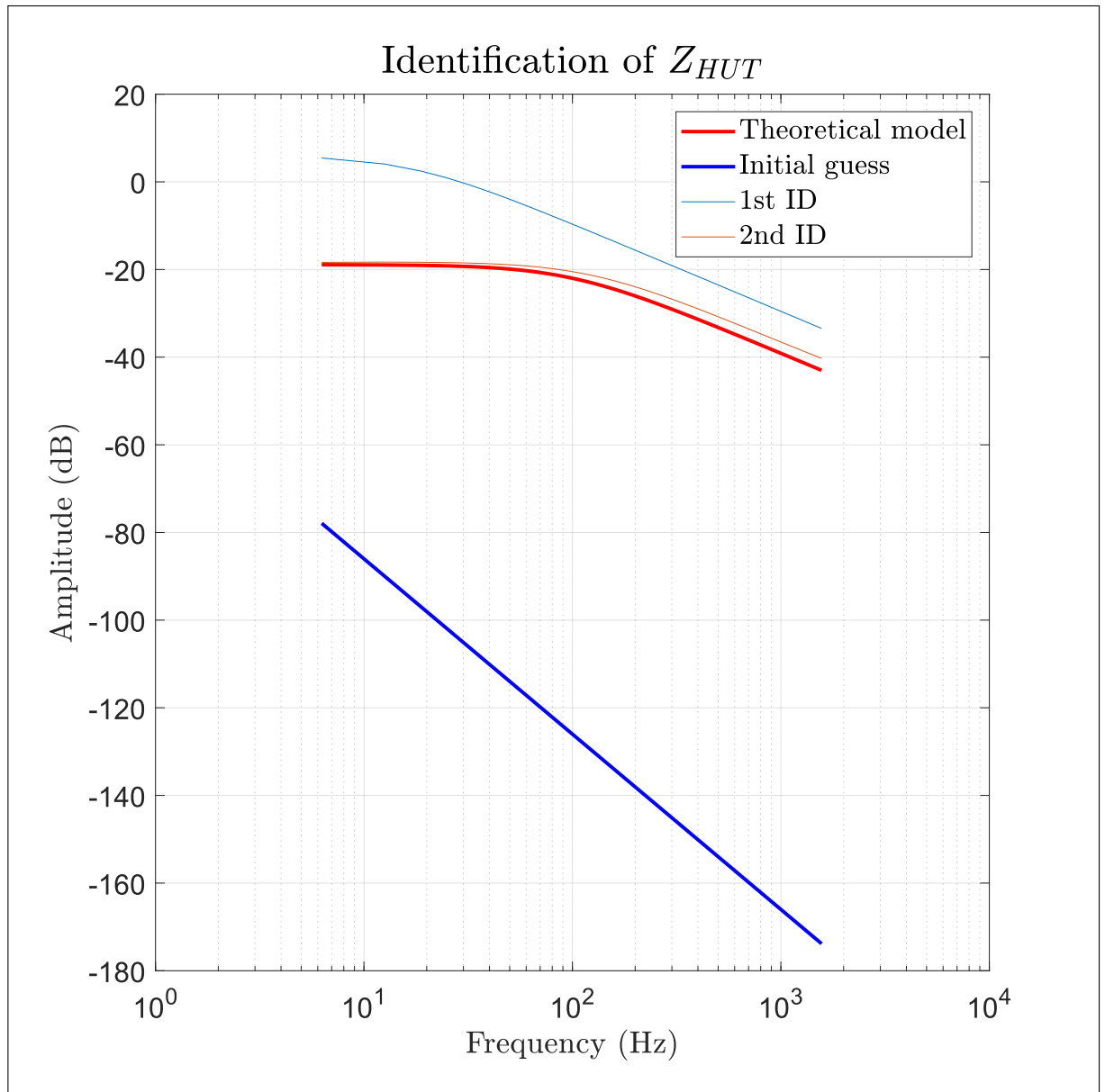


FIGURE 4.13. Identification process. First 2 attempts

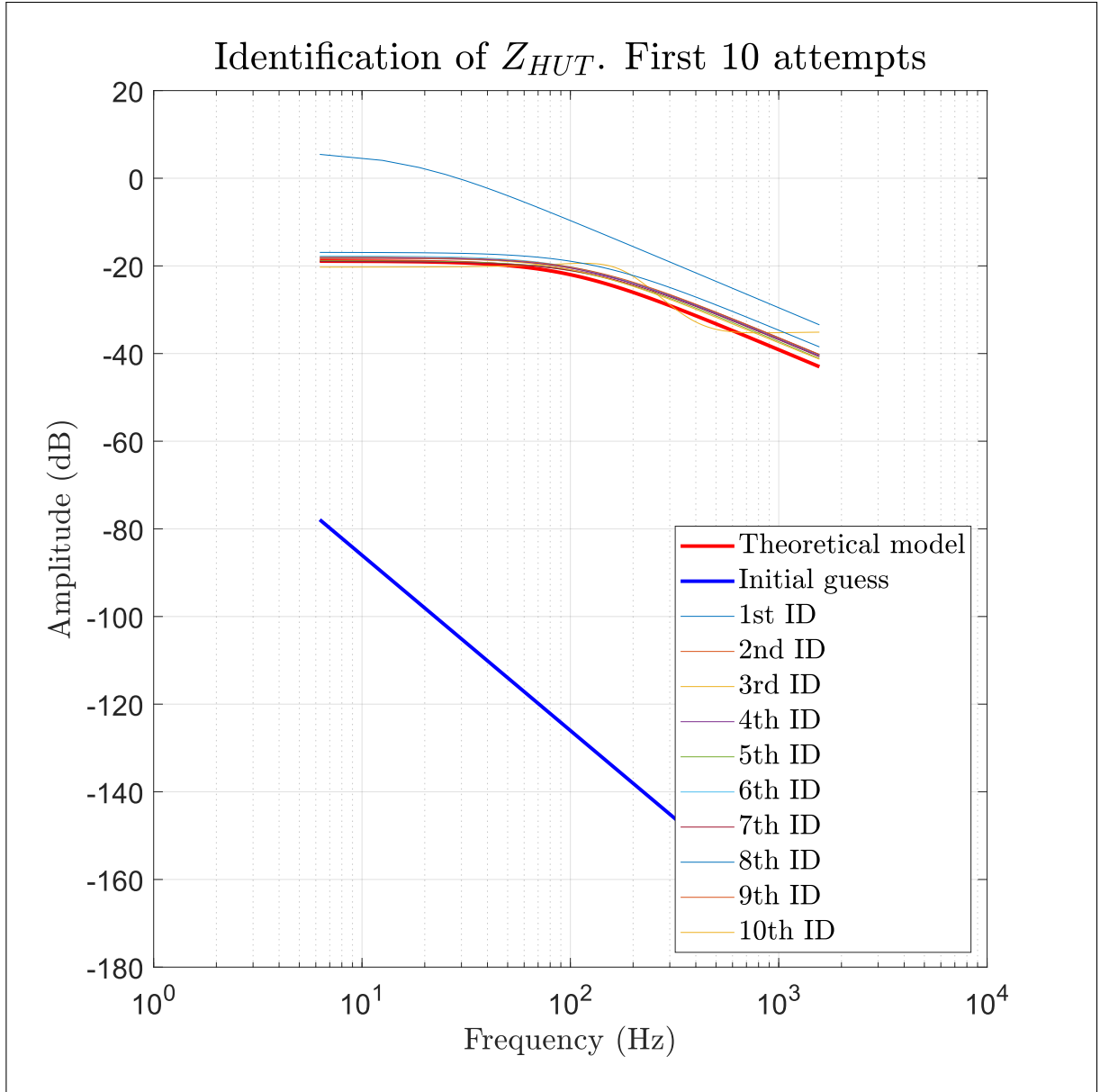


FIGURE 4.14. Identification process. First 10 attempts

The initial miss identification of Z_{HUT} is also visible in the behaviour of all electrical variables. Figures 4.15, 4.16, 4.17 and 4.18 show how the voltage, current, active power and reactive power behaved during the simulation. We see how it isn't until second 6,

when Z_{HUT} is identified successfully, that the variables of the software side and hardware side equalise and their values converge towards those of the reference circuit (NCS).

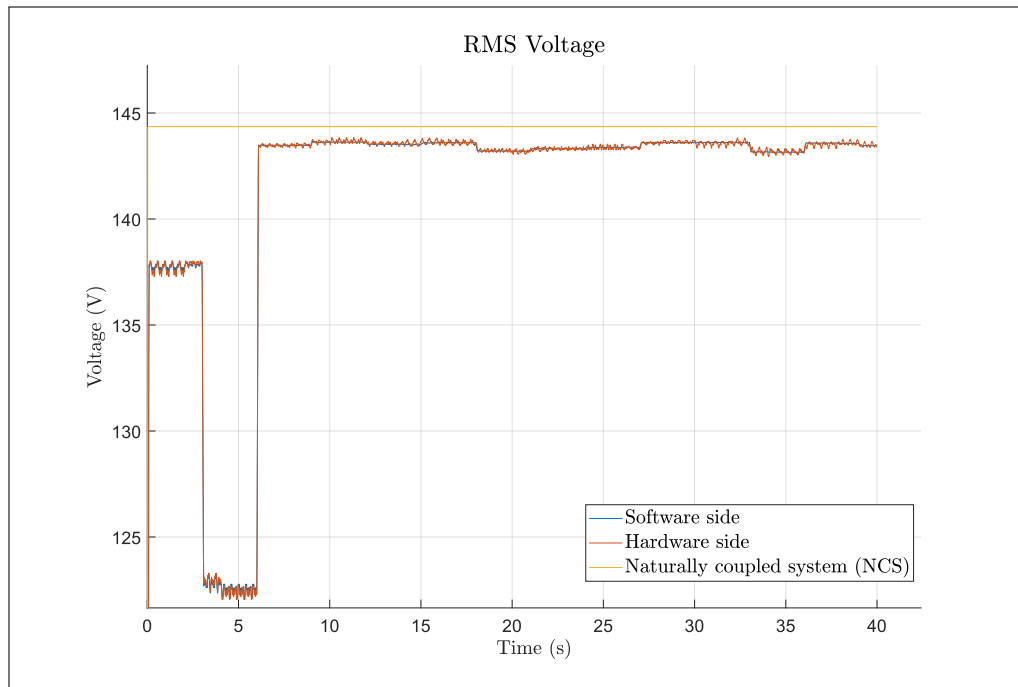


FIGURE 4.15. Simulation voltage. RL load

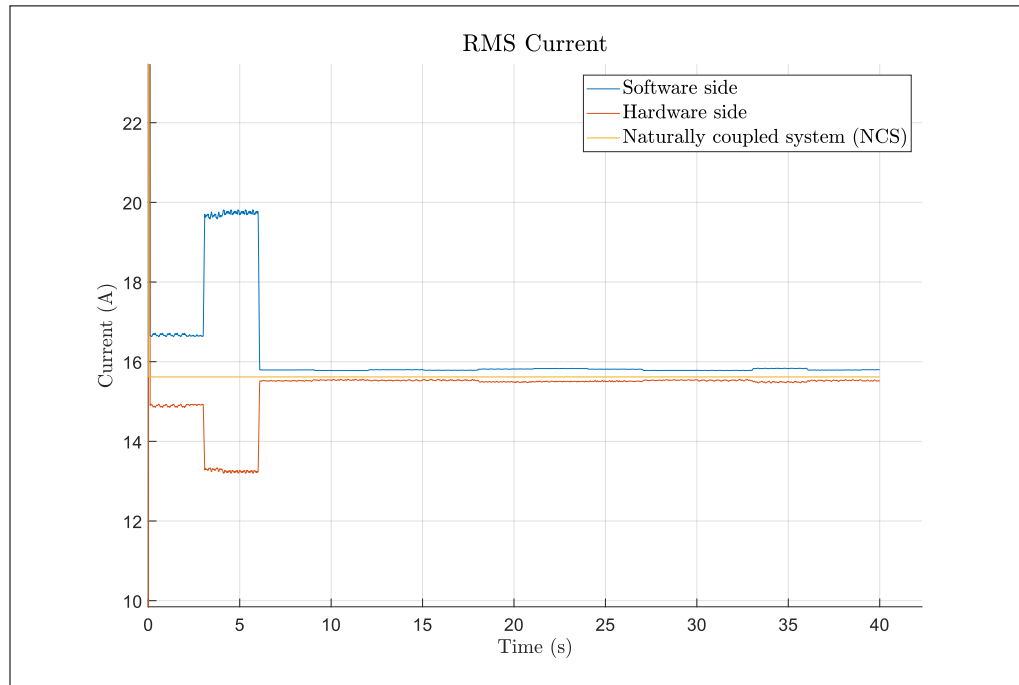


FIGURE 4.16. Simulation current. RL load

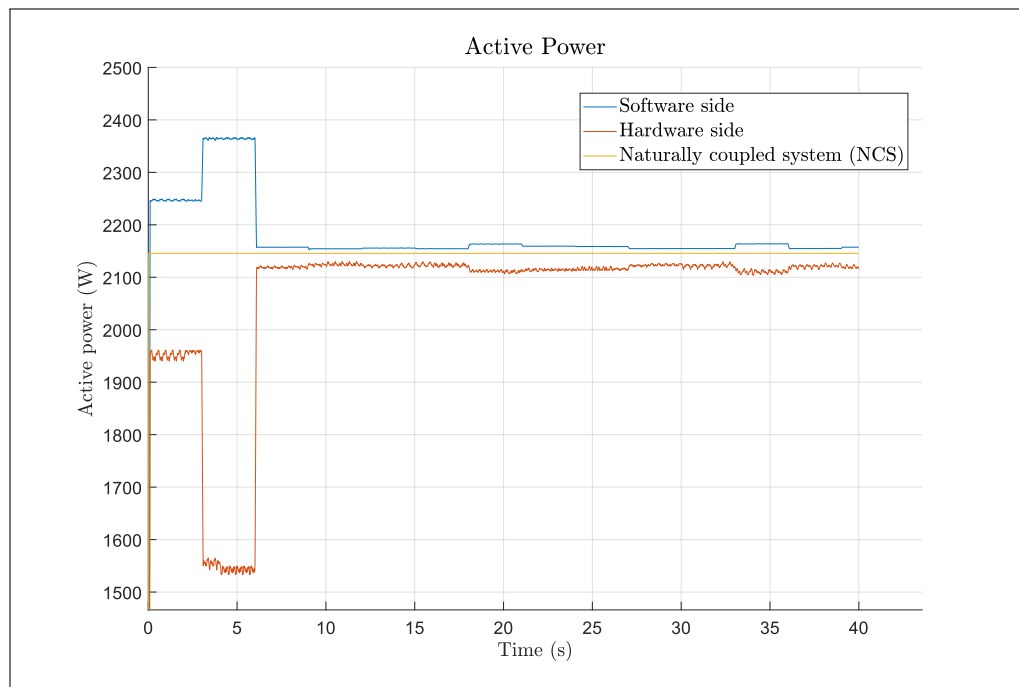


FIGURE 4.17. Active power. RL load

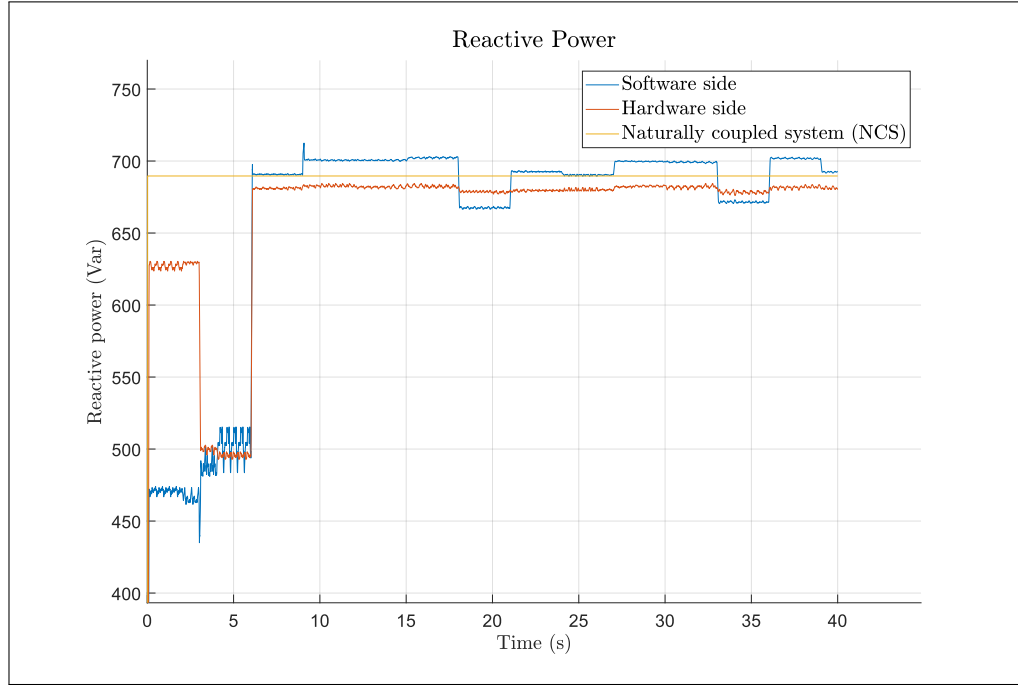


FIGURE 4.18. Reactive power. RL load

Figures 4.19 and 4.20 show the percentage of error with respect to the naturally coupled system (NCS), of the software side and hardware side variables. Voltage is the most accurate variable with average relative errors with respect to the reference circuit of less than 0.9 %.

Figures 4.21 and 4.22 show that current behaves similarly well with average error between 0.5 % and 1 % for the software side and 1 to 1.5 % for hardware side, once the identification is complete.

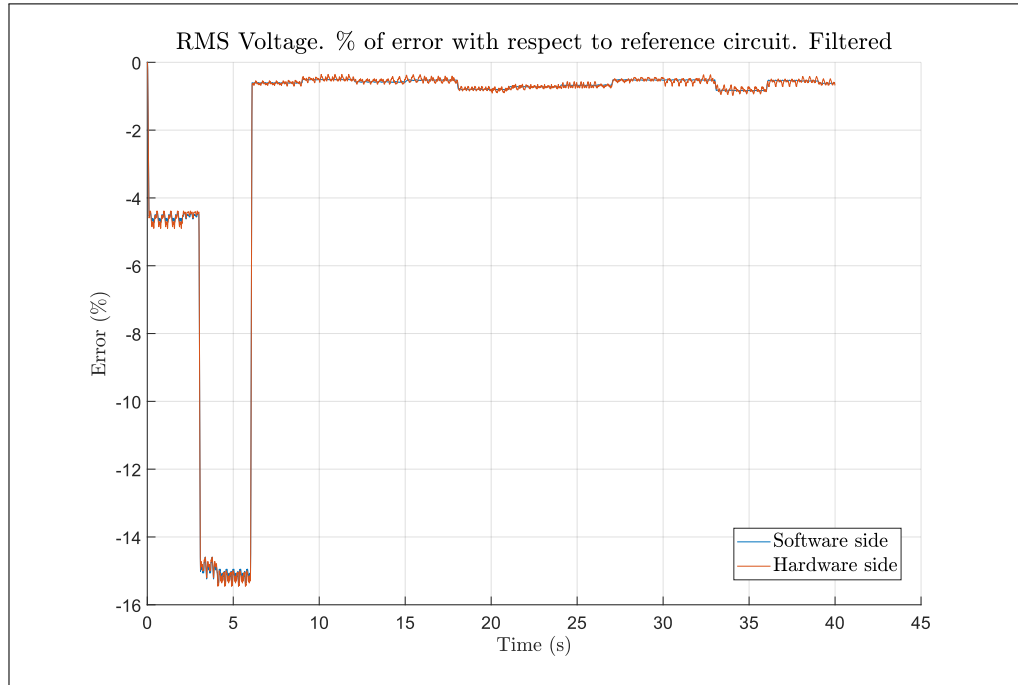


FIGURE 4.19. Voltage relative error. RL load

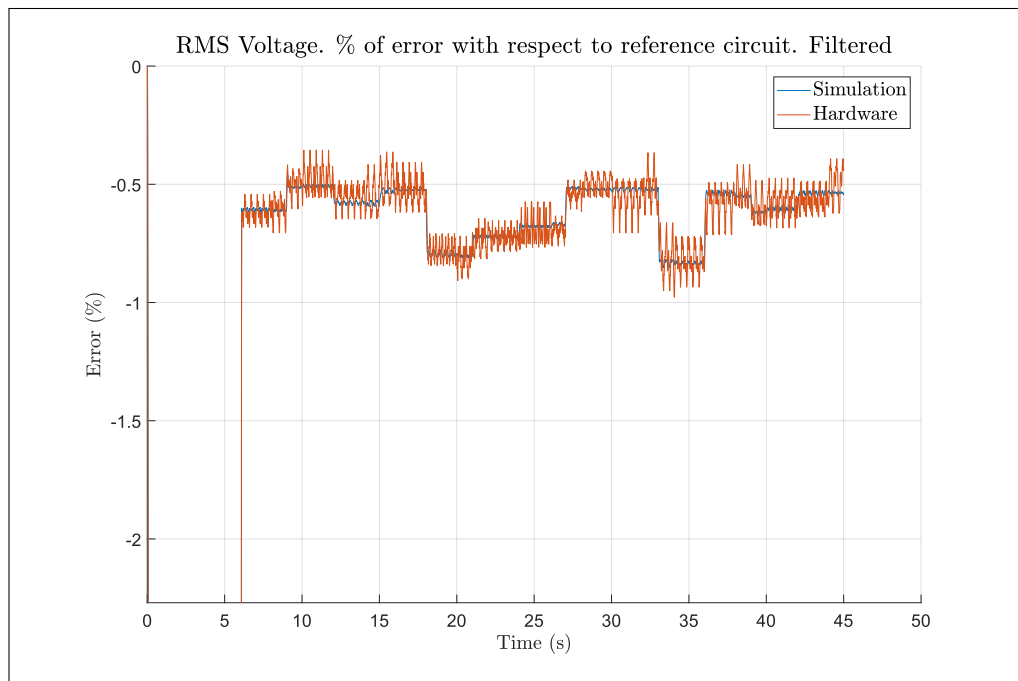


FIGURE 4.20. Voltage relative error zoom. RL load

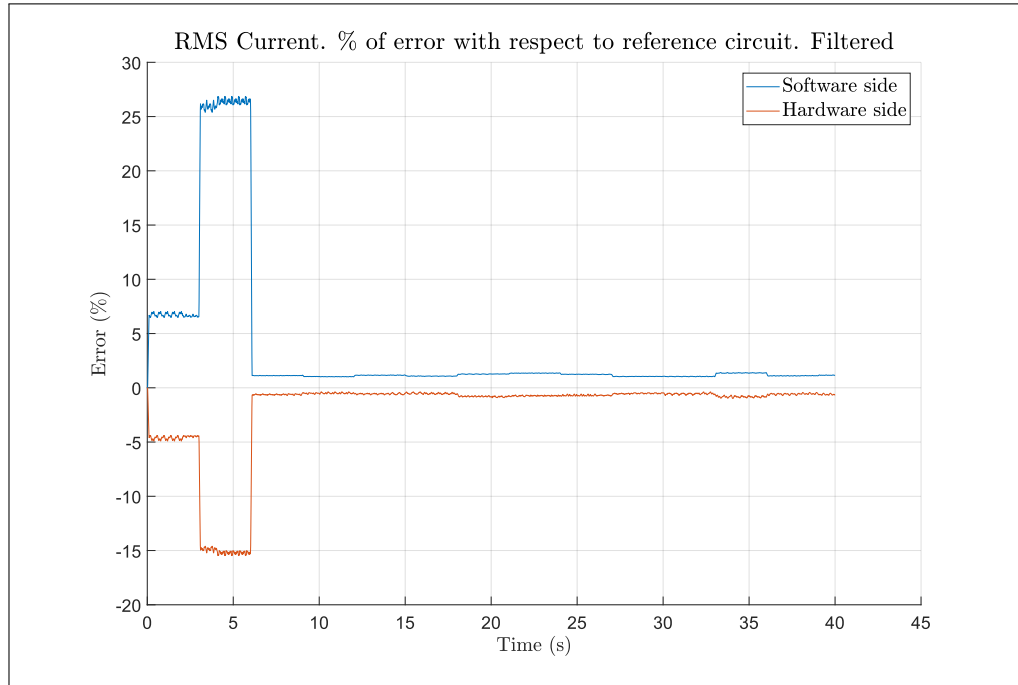


FIGURE 4.21. Current relative error. RL load

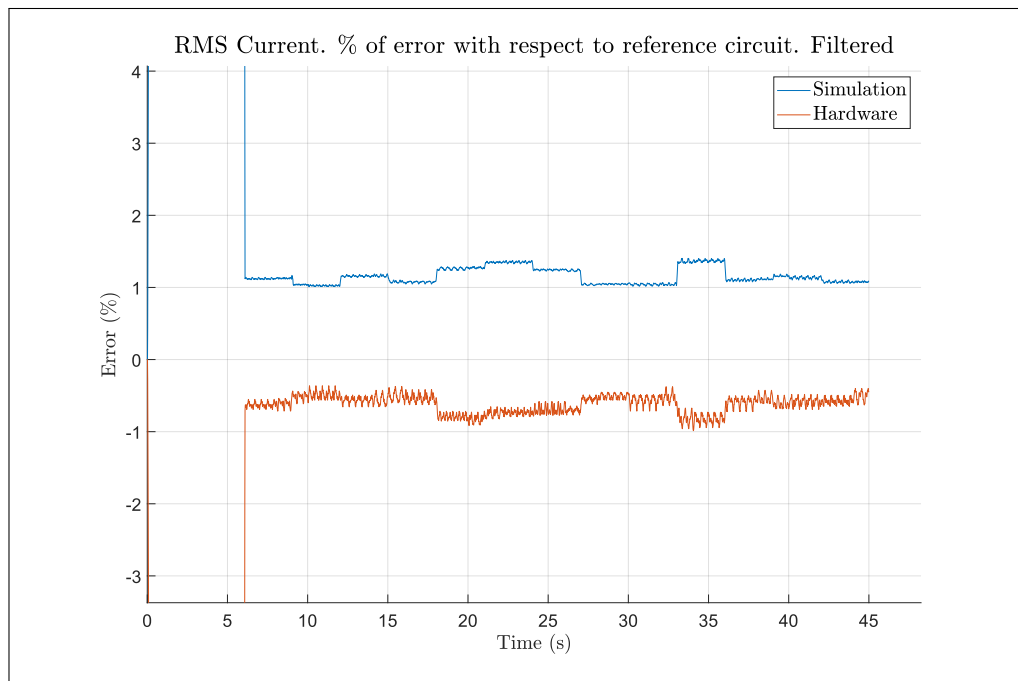


FIGURE 4.22. Current relative error zoom. RL load

Active power also behaved well, with average errors of 0.5 % to 1 % in the software side and 1 to 2 % in the hardware side, as seen in figure 4.23 and 4.24. Finally, the results for reactive power can be seen in figure 4.25 and 4.26. Reactive power is usually the hardest variable to obtain accurately, as in most cases, it tends to be small when compared to active power. In these situations, a small error in the phase between the voltage and current will lead to very little change in active power, specially in relative terms, but to substantial changes in reactive power. Thus small inaccuracies in the experiment will disproportionately affect the results for reactive power, leading to high relative error. The opposite would be true, in a scenario where reactive power is much greater than active power. Here, active power would be subject to higher degrees of error.

Despite the above, results show an average errors of 2 to 3 % in the software side while the hardware side shows error of less than 2 % with respect to the reference. Overall, this is comparable with results from and after applying compensation strategies.

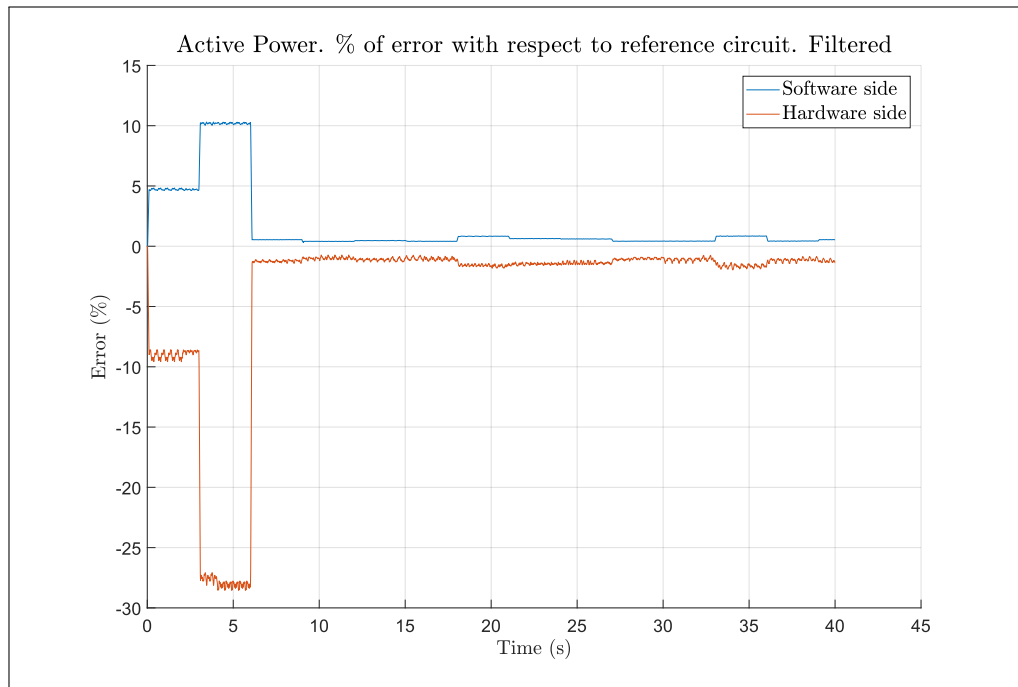


FIGURE 4.23. Active power relative error. RL load

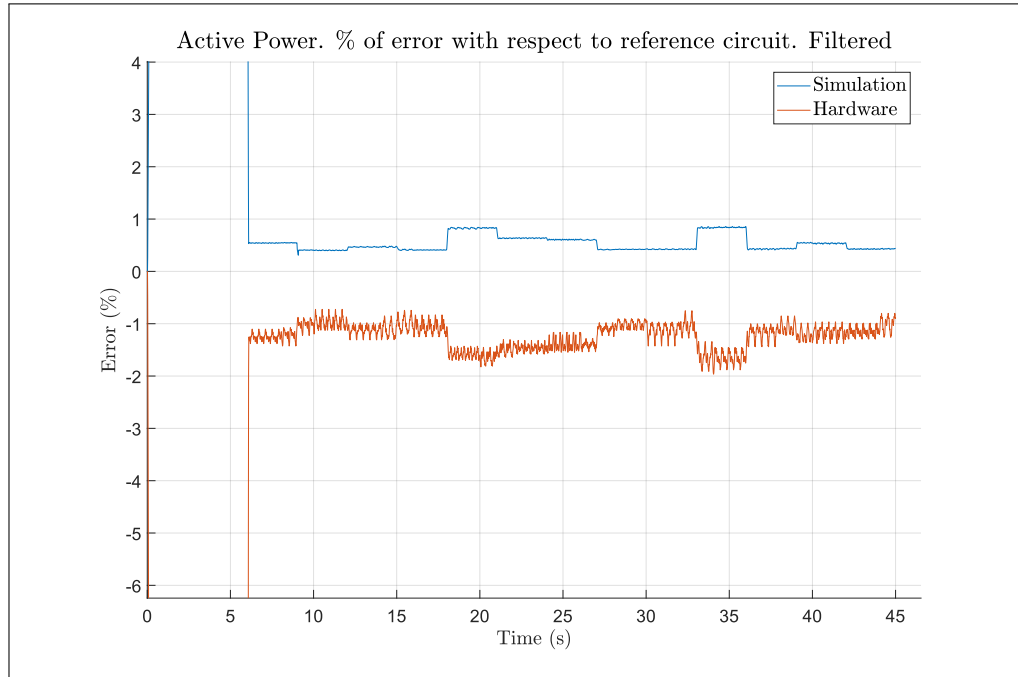


FIGURE 4.24. Active power relative error zoom. RL load

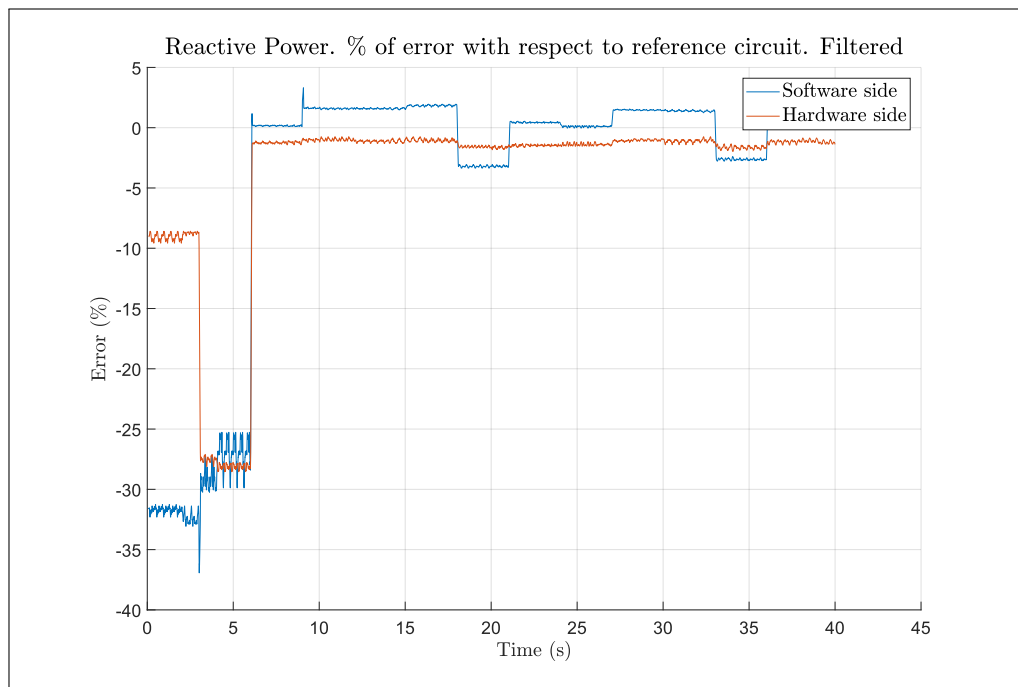


FIGURE 4.25. Reactive power relative error. RL load

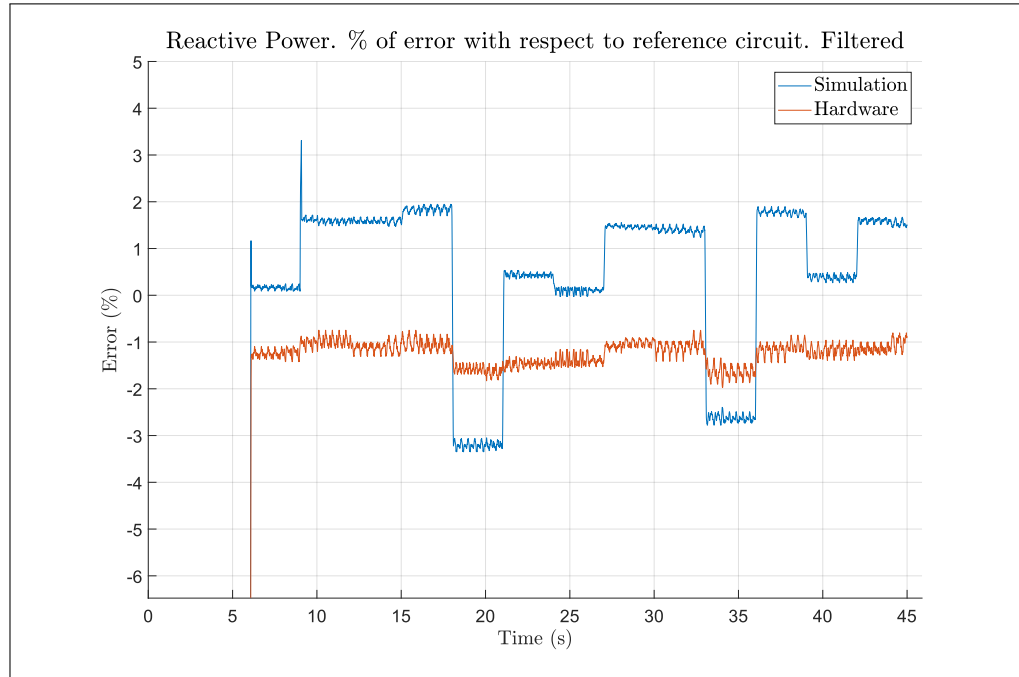


FIGURE 4.26. Reactive power relative error zoom. RL load

4.6.2. Active load

For this scenario the RL filter in the previous simulation was replaced with a grid tied inverter, with a simple L output filter. For convenience, the inverter was simulated using an average model as detailed electrical simulations with transistors require very short timesteps and would not contribute relevant information to the experiments, in this context. The only subsystem that saw changes with respect to the previous experiment was SM_computations. Figure 4.27 shows this subsystem and how the simulated inverter was integrated.

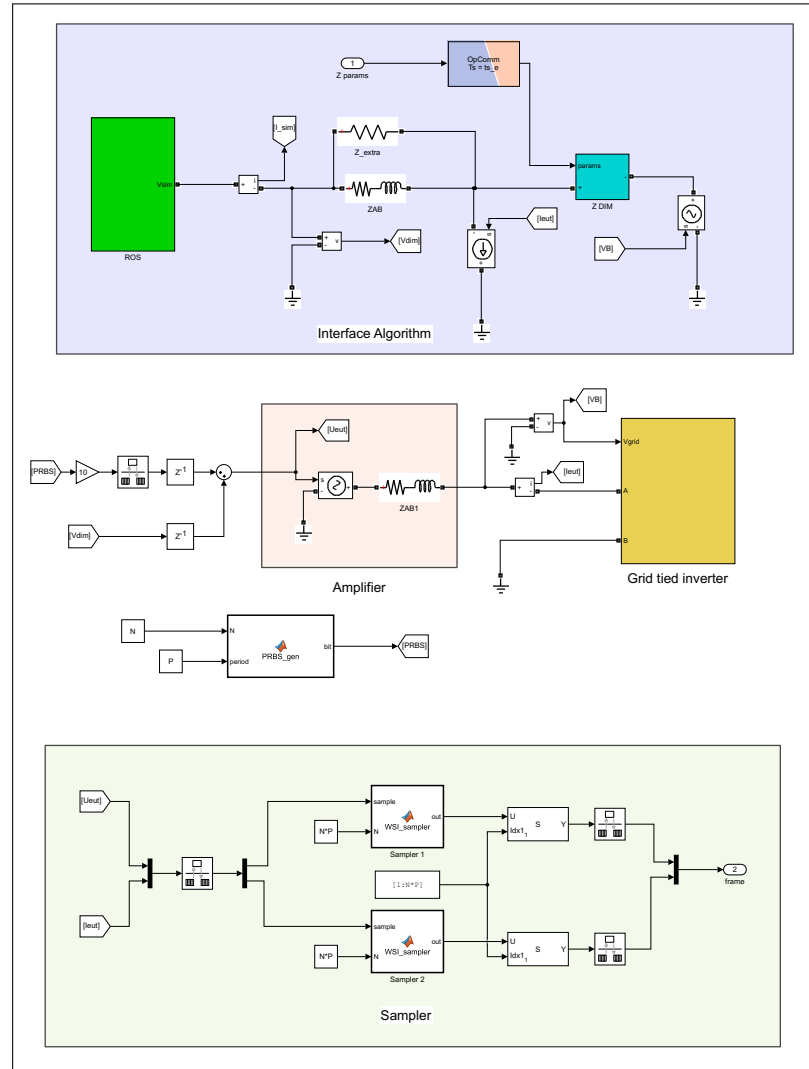


FIGURE 4.27. SM_computations subsystem used for simulation with inverter

Figure 4.28 shows the model used for the inverter. A PLL is used to provide a voltage reference synchronised with the grid as well as its angle. The current control, shown in figure 4.29, is implemented using two PI controllers in the dq axis. To achieve this on a single phase system, a time delay is applied to the current to create a second virtual phase with a 90° shift. Then, we apply the $\alpha\beta_0/dq0$ transformation to obtain the currents in the d and q axis. A PI controller makes sure the currents in both axis meet their references with

minimal error. Finally, we revert back to the stationary reference frame by applying the inverse of the previous transformation ($dq0/\alpha\beta0$). From here, we use only the α component as the voltage reference for the inverter (U_{ref}).

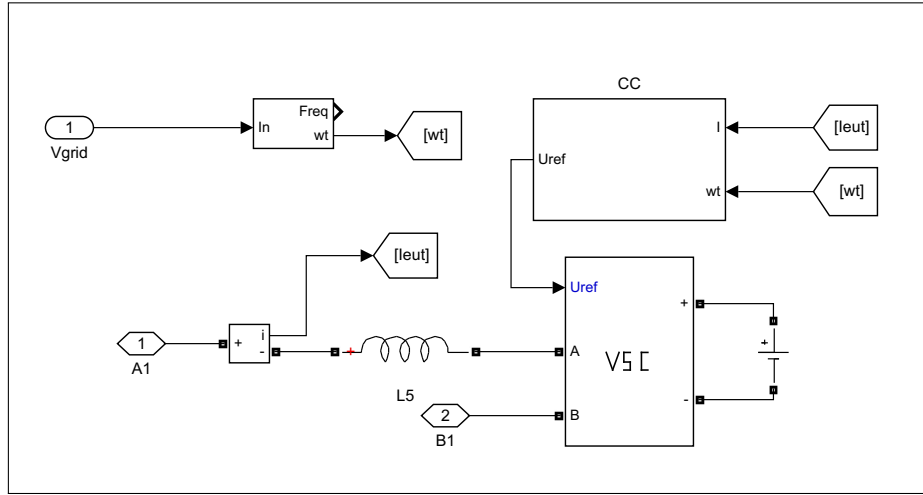


FIGURE 4.28. Grid tied inverter

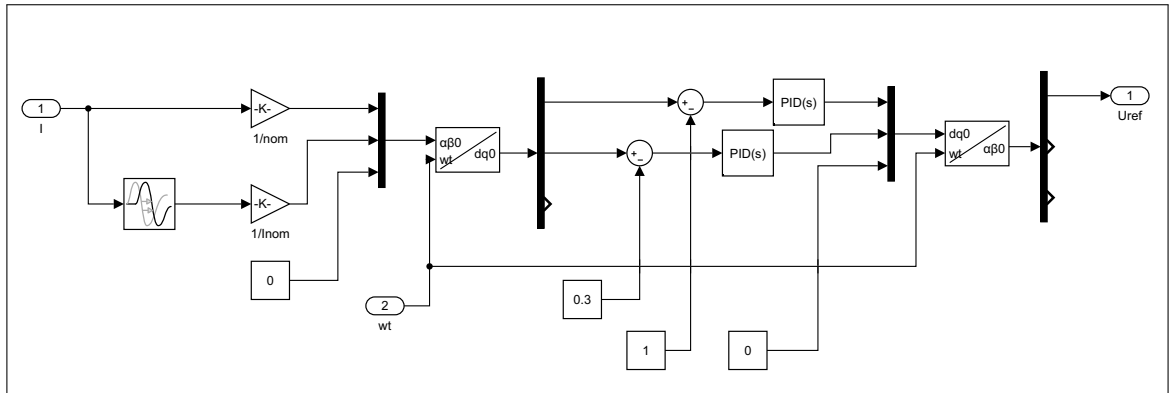


FIGURE 4.29. Current control scheme

A simplified diagram of the simulation is shown in figure 4.30. As in the previous case, the software side consist on a $220V_{RMS}$ AC source and a series impedance Z_s . The inverter is powered by a 400VDC source. Further relevant parameters can be found in table 4.2.

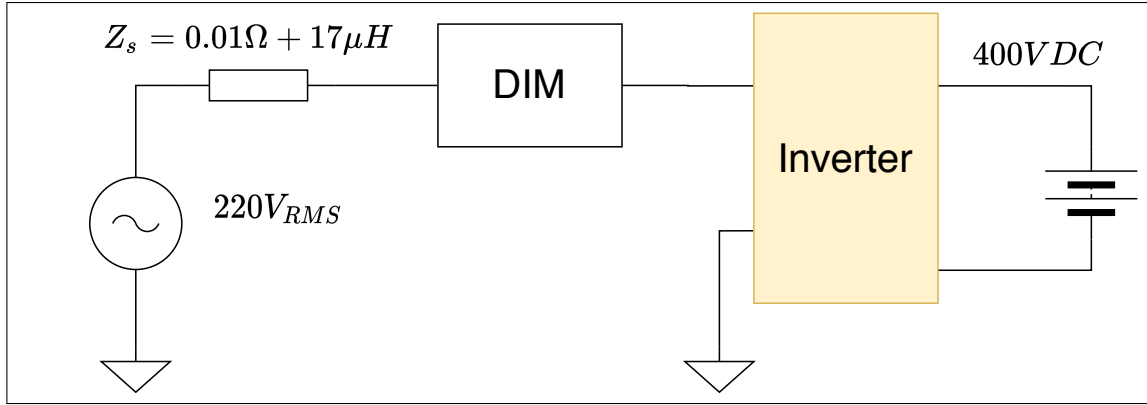


FIGURE 4.30. Simplified diagram of the simulation

TABLE 4.2. Simulation parameters: Inverter load

Parameter	Value
N	2000
P	5
Kp	14.7027
Ki	62.8319
Window size (γ)	250
SM_computations timestep	$100\mu s$
SS_RMS timestep	$200\mu s$
SS_ID timestep	$2s$
Software impedance (Z_s)	$0,01\Omega + 17\mu H$

The experiment with the inverter did not provide satisfactory results. While variables on the hardware side are very accurate with relative error being close to zero, results for active and reactive power on the simulation side are consistently biased and display relatively large error. Figures 4.31, 4.33, 4.36 and 4.39 show how the voltage, current, active and reactive power behaved during the simulation. We see that while the hardware side was very accurate, the results in the software side were consistently biased. Results

for active power show a relative error between 4 and 6 % as seen in figure 4.38 once the impedance is identified. However the biggest problems lie with the reactive power where relative error reaches between -7 % and -15 % 4.41. Such large error imply there is a problem with our approach and indeed there is.

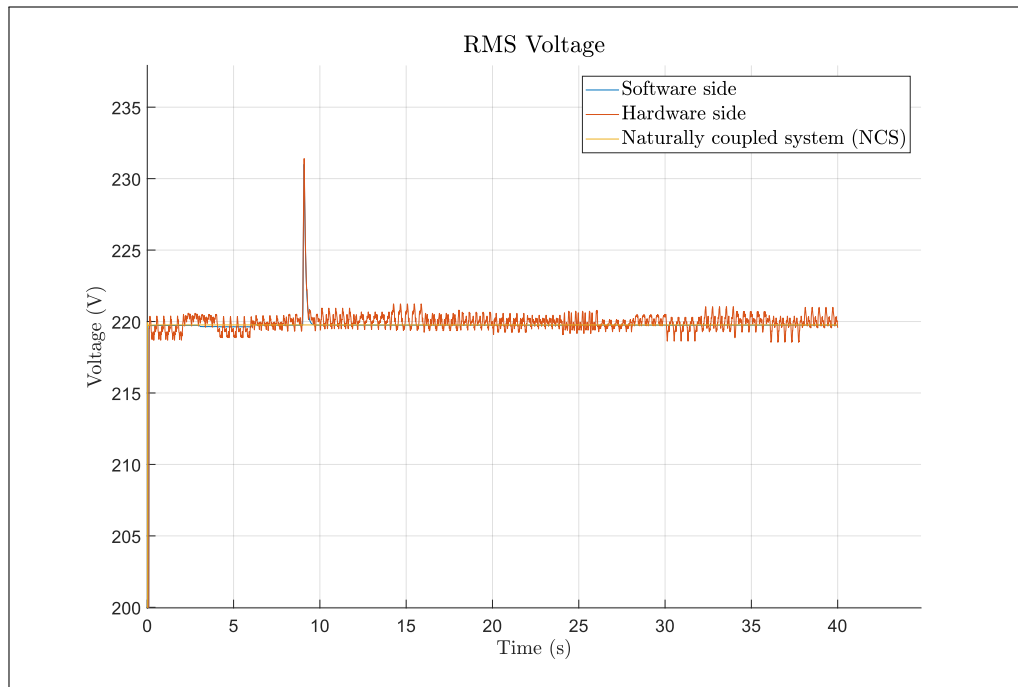


FIGURE 4.31. RMS voltage. Inverter load

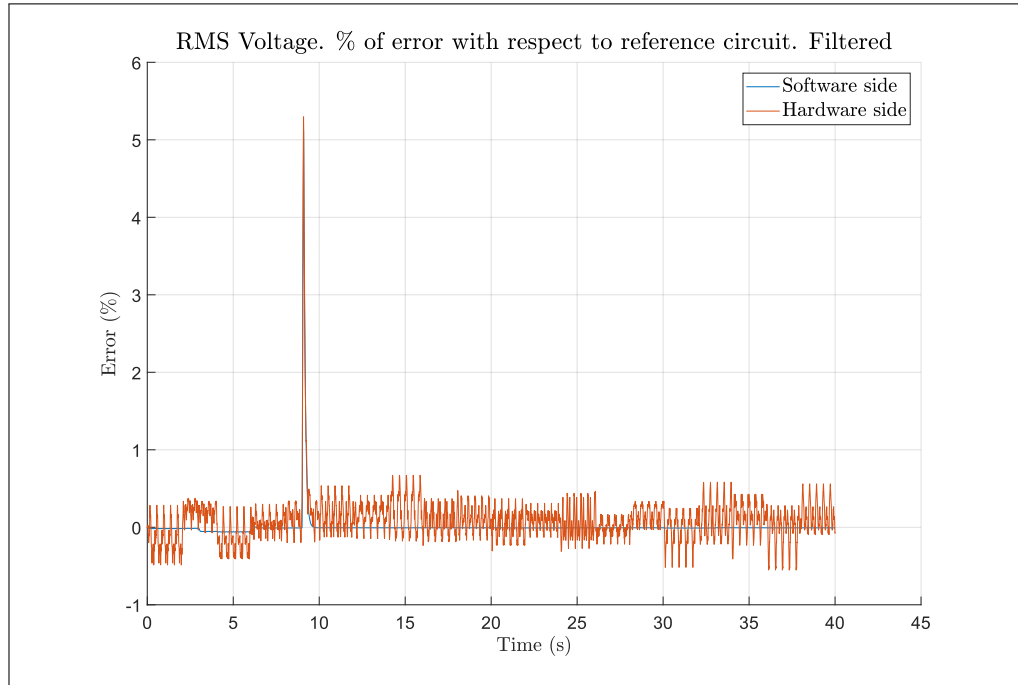


FIGURE 4.32. Voltage relative error. Inverter load

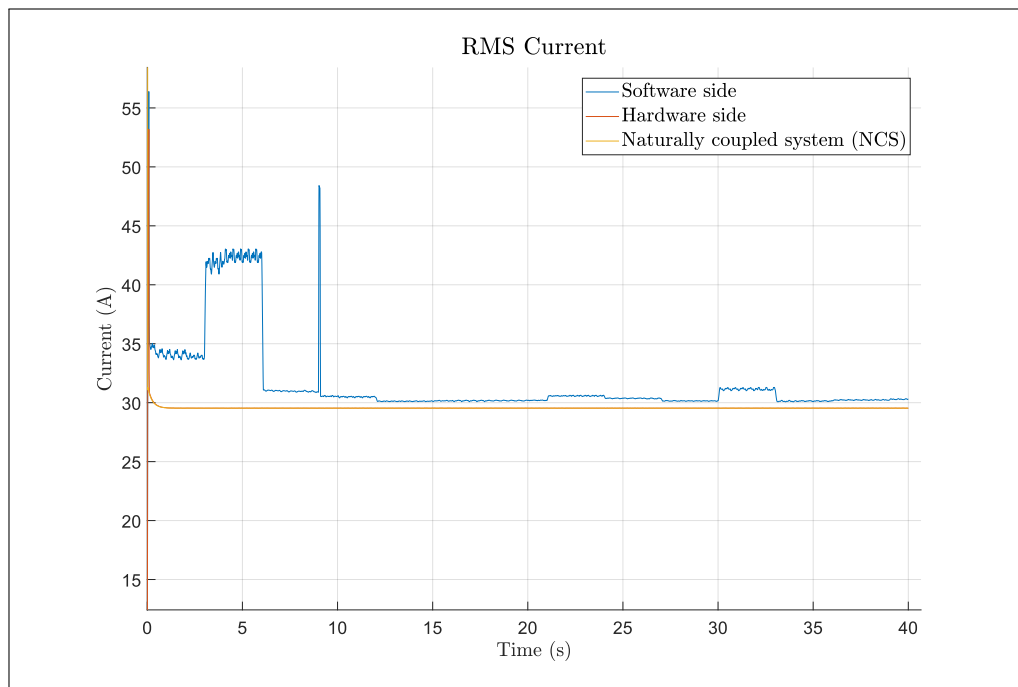


FIGURE 4.33. RMS current. Inverter load

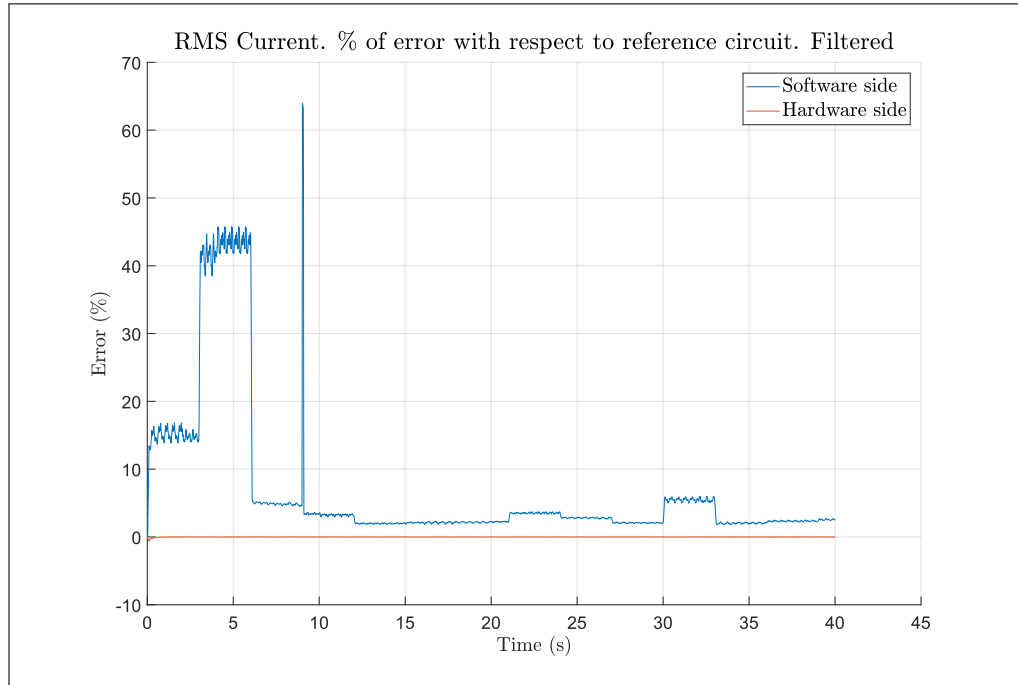


FIGURE 4.34. Current relative error. Inverter load

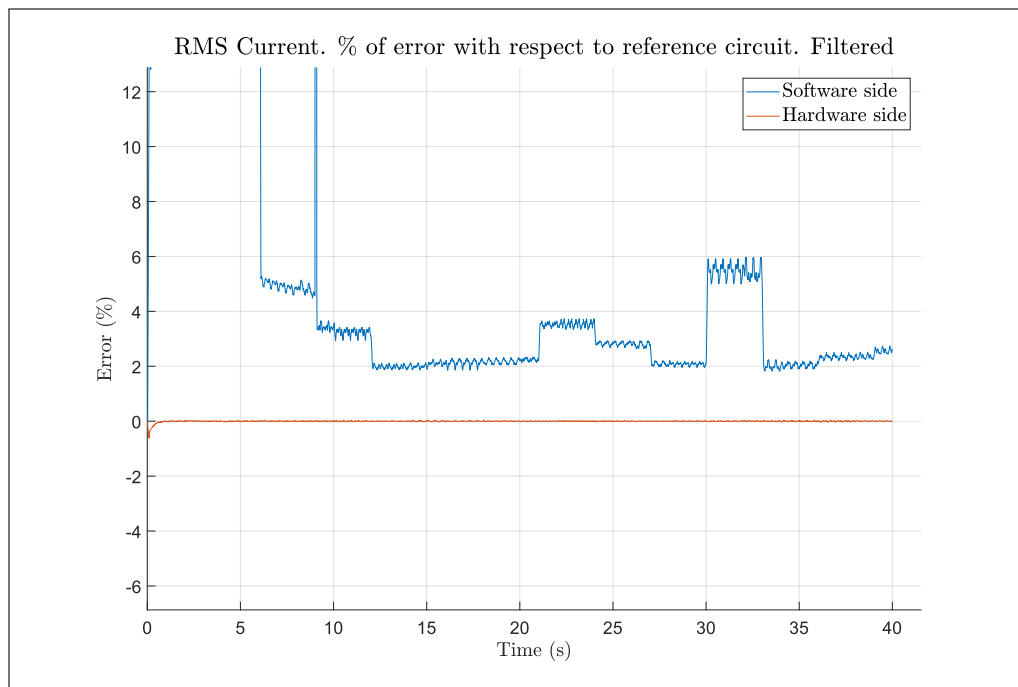


FIGURE 4.35. Current relative error zoom. Inverter load

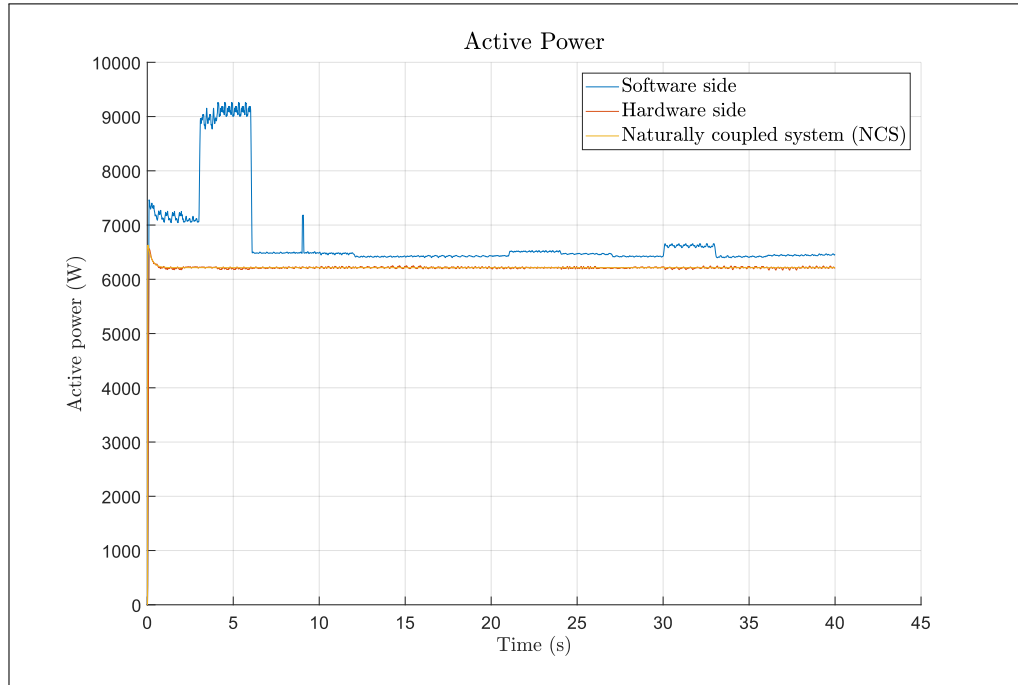


FIGURE 4.36. Active power. Inverter load

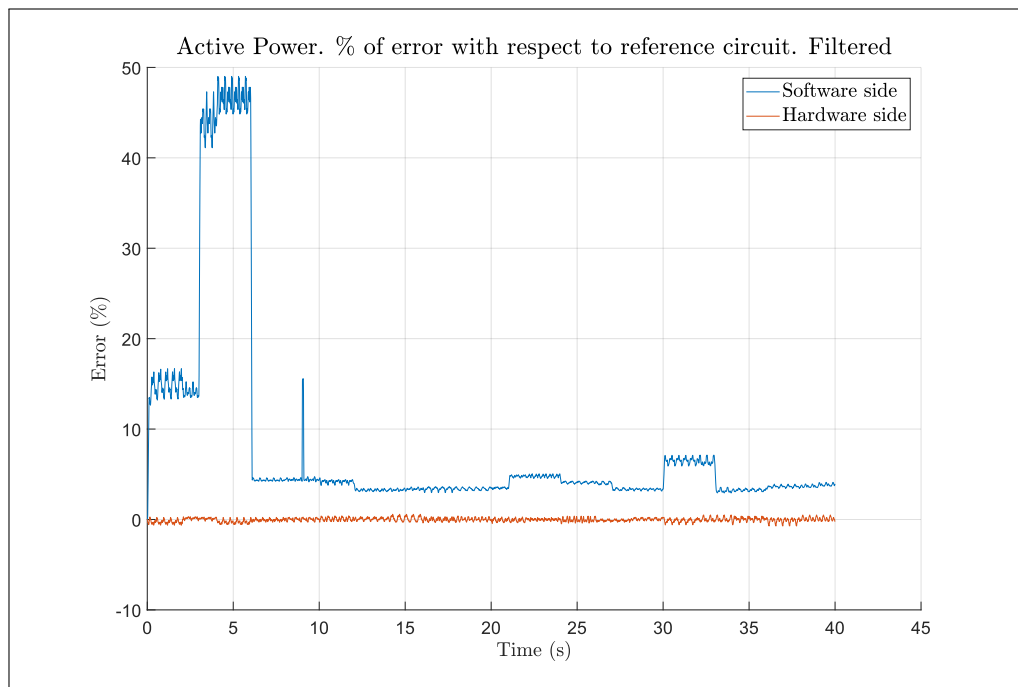


FIGURE 4.37. Active power relative error. Inverter load

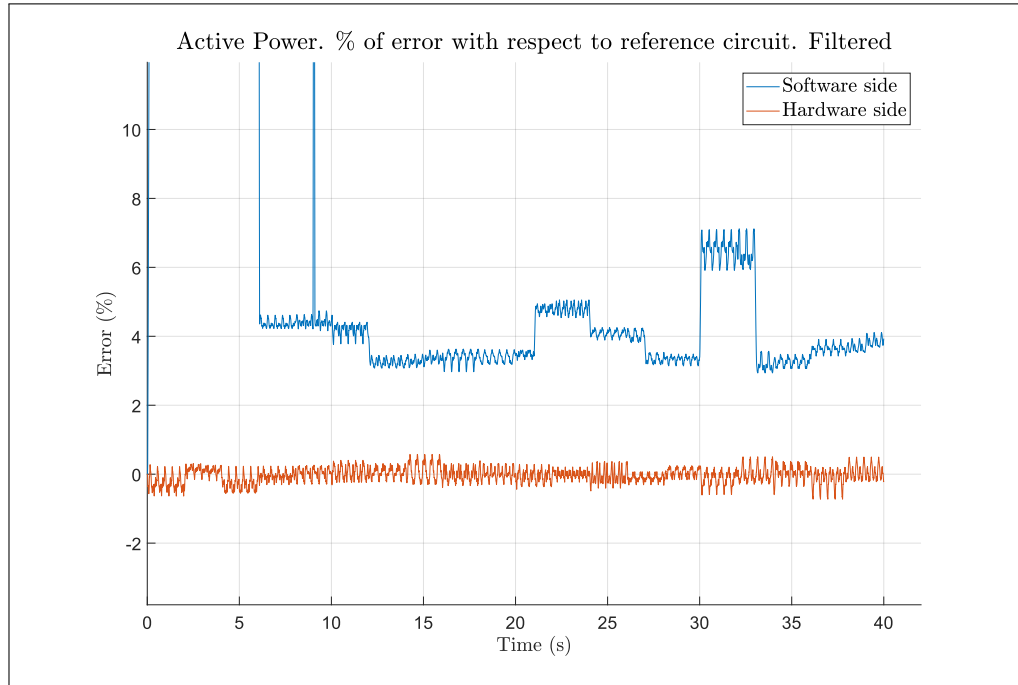


FIGURE 4.38. Active power relative error zoom. Inverter load

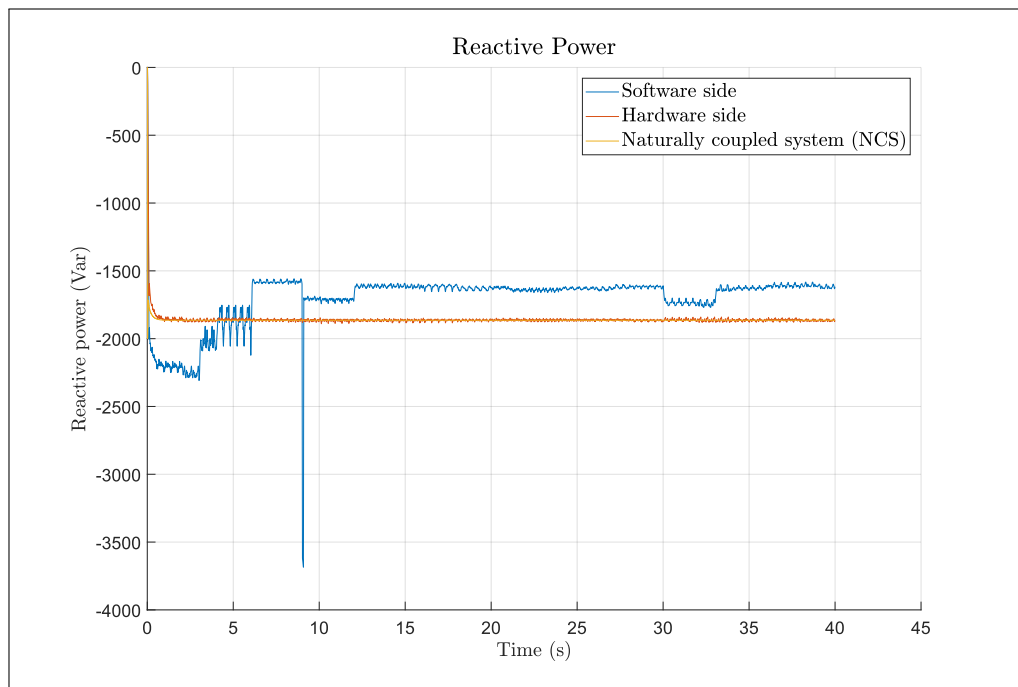


FIGURE 4.39. Reactive power. Inverter load

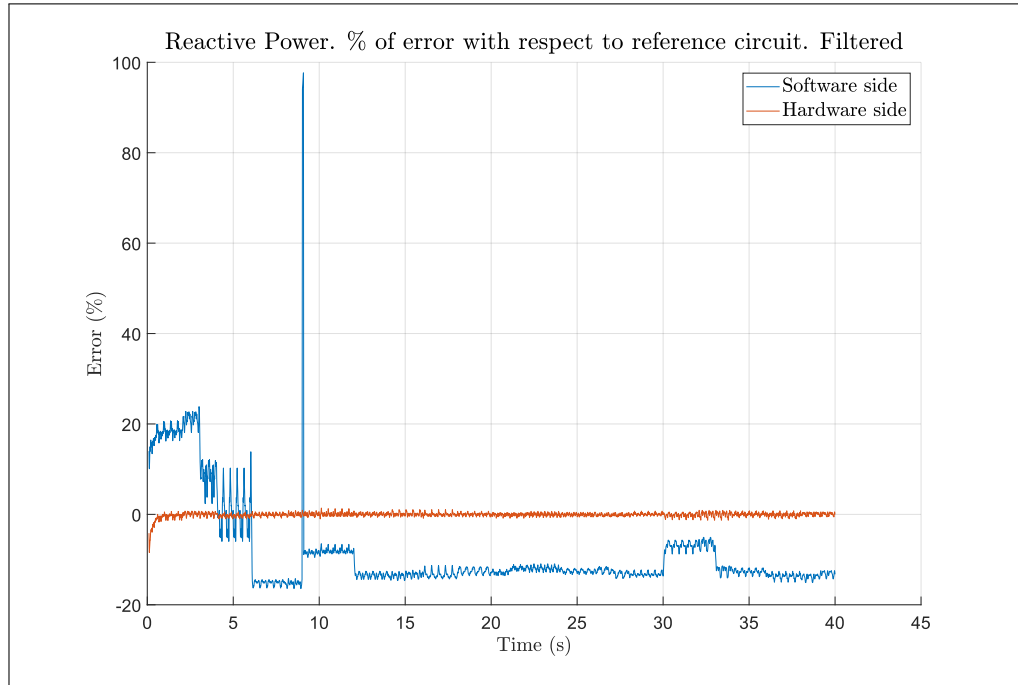


FIGURE 4.40. Reactive power relative error. Inverter load

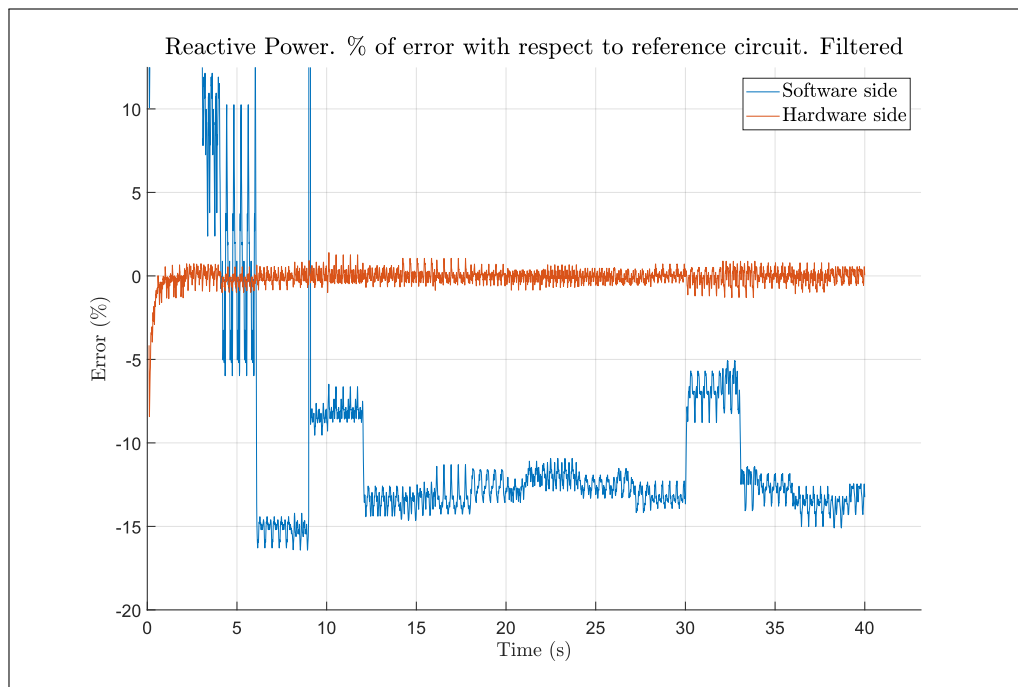


FIGURE 4.41. Reactive power relative error zoom. Inverter load

4.7. Problems of spectral analysis in the active load case

The problem shown in previous section can be explained due to a poor identification of Z_{HUT} . Unlike the case for the passive load, where spectral analysis proved useful, we will see that for this case the estimates we obtain will be biased, making this technique unfit for identifying HUT such as our inverter. We will explore the reason behind this and confirm it repeating the experiment in section with the real PHIL platform and a real inverter. We will then present a small work around that both support our claim about why spectral analysis fails while partially solving the issue with it.

An explanation for the problem with the experiment above can be found in Chapter 10 of *Systems identification* by [Torsten y Stoica \(1989a\)](#), which explains the concerns and challenges one must consider when identifying systems under closed-loop operation. The problem lays in the fact that as our HUT now operates as a closed-loop system, there is a feedback path that affects the dynamics between $u(t)$ and $y(t)$. More precisely the inverter case can be modelled as seen in figure 4.42, where q^{-1} is the shift operator.

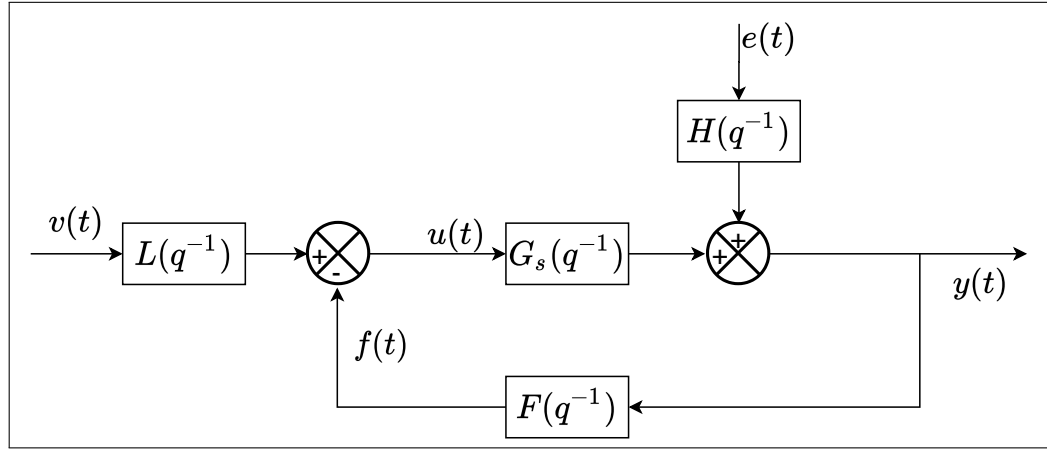


FIGURE 4.42. Model of a feedback system

For this type of system, we get that we can calculate $u(t)$ and $y(t)$ considering the feedback as follows. w

$$y(t) = [I + G_s(q^{-1})F(q^{-1})]^{-1}[G_s(q^{-1})L(q^{-1})v(t) + H(q^{-1})e(t)] \quad (4.1)$$

$$\begin{aligned} u(t) = & [L(q^{-1}) - F(q^{-1})(I + G_s(q^{-1})F(q^{-1}))^{-1}G_s(q^{-1})L(q^{-1})]v(t) \\ & - F(q^{-1})(I + G_s(q^{-1})F(q^{-1}))^{-1}H(q^{-1})e(t) \end{aligned} \quad (4.2)$$

Now taking the SISO case and assuming the feedforward part as $L(q^{-1}) \equiv 1$ for simplicity, we can reduce equations 4.1 and 4.2 into the following:

$$z(t) = F(q^{-1})H(q^{-1})e(t) \quad (4.3)$$

$$y(t) = \frac{1}{1 + G_s(q^{-1})F(q^{-1})}[G_s(q^{-1})v(t) - \frac{1}{F(q^{-1})}z(t)] \quad (4.4)$$

$$u(t) = \frac{1}{1 + G_s(q^{-1})F(q^{-1})}[v(t) + z(t)] \quad (4.5)$$

Calculating the spectrum Φ_{yu} and Φ_u we obtain equations

$$\Phi_{yu}(\omega) = \frac{1}{|1 + G_s(e^{-i\omega})F(e^{-i\omega})|^2}[G_s(e^{-i\omega})\Phi_v(\omega) - \frac{1}{F(e^{-i\omega})}\Phi_z(\omega)] \quad (4.6)$$

$$\Phi_u(\omega) = \frac{1}{|1 + G_s(e^{-i\omega})F(e^{-i\omega})|^2}[\Phi_v(\omega) + \Phi_z(\omega)] \quad (4.7)$$

Then $\hat{G}(e^{-i\omega})$ can be calculated as in equation 3.33.

$$\hat{G}(e^{i\omega}) = \frac{\Phi_{yu}}{\Phi_u} = \frac{G_s(e^{-i\omega})\Phi_v(\omega) - \frac{1}{F(e^{-i\omega})}\Phi_z(\omega)}{\Phi_v(\omega) + \Phi_z(\omega)} \quad (4.8)$$

From equation 4.8 we can quickly spot that if $(z(t) = 0, \Phi_z(\omega) = 0)$ then the estimates gives exactly the dynamic of $G_s(e^{-i\omega})$.

$$\hat{G}(e^{-i\omega}) = G_s(e^{-i\omega}) \quad (4.9)$$

Since $z(t) = F(q^{-1})H(q^{-1})e(t)$. This leaves us with three possibilities: $F(q^{-1}) = 0$, $H(q^{-1}) = 0$, or $e(t) = 0$. Since all real systems will have some source of noise and we have seen in section 3.5 and in simulation that systems where $H(q^{-1}) \neq 0$ can be properly identified, only $F(q^{-1})$, the feedback dynamic remains as possible culprit.

To verify this, we can calculate the difference between $\hat{G}(e^{-i\omega})$ and $G_s(e^{-i\omega})$ in the general case. This, we can calculate using 4.8 and obtain 4.10.

$$\hat{G}(e^{-i\omega}) - G_s(e^{-i\omega}) = -\frac{\Phi_z(\omega)}{\Phi_z(\omega) + \Phi_v(\omega)} \frac{1 + G_s(e^{-i\omega})F(e^{-i\omega})}{F(e^{-i\omega})} \quad (4.10)$$

Substituting for the spectrum of $z(t)$, $\Phi_z(\omega) = |F(e^{-i\omega})|^2 |H(e^{-i\omega})|^2 \Phi_e(\omega)$ in 4.10 and evaluating a $F(e^{-i\omega}) = 0$ results in the difference between the real dynamic and the estimate is 0.

$$\hat{G}(e^{-i\omega}) - G_s(e^{-i\omega}) = -\frac{|F(e^{-i\omega})|^2 |H(e^{-i\omega})|^2 \Phi_e(\omega)}{|F(e^{-i\omega})|^2 |H(e^{-i\omega})|^2 \Phi_e(\omega) + \Phi_v(\omega)} \frac{1 + G_s(e^{-i\omega})F(e^{-i\omega})}{F(e^{-i\omega})} \quad (4.11)$$

$$\hat{G}(e^{-i\omega}) - G_s(e^{-i\omega}) = -\frac{|F(e^{-i\omega})| |H(e^{-i\omega})|^2 \Phi_e(\omega)}{|F(e^{-i\omega})|^2 |H(e^{-i\omega})|^2 \Phi_e(\omega) + \Phi_v(\omega)} (1 + G_s(e^{-i\omega})F(e^{-i\omega})) \quad (4.12)$$

$$\hat{G}(e^{-i\omega}) - G_s(e^{-i\omega}) = -\frac{0 \cdot |H(e^{-i\omega})|^2 \Phi_e(\omega)}{0 \cdot |H(e^{-i\omega})|^2 \Phi_e(\omega) + \Phi_v(\omega)} (1 + G_s(e^{-i\omega}) \cdot 0) \quad (4.13)$$

$$\hat{G}(e^{-i\omega}) - G_s(e^{-i\omega}) = 0 \quad (4.14)$$

This means that only in open loop operation the system will be identifiable through spectral analysis as $F(q^{-1}) \equiv 0$. For cases where there is feedback, the technique will give biased estimates. This is a problem as our active HUT is indeed a closed loop system.

As such, spectral analysis is likely not suitable for identifying active HUT while these are powered on.

In fact, upon further research the proper identification of the output impedance of power devices appears to pose several challenges. According to [Ljung \(1999\)](#) and referencing to figure 4.42, when dealing with closed loop systems there are 3 approaches for identification:

- *Direct approach:* Proceed as if the system operates in an open loop and use $u(t)$ and $y(t)$ to identify the system.
- *Indirect approach:* use $v(t)$ and $y(t)$ to identify the close loop dynamic and the use prior knowledge of $F(q^{-1})$ to calculate $G_s(q^{-1})$.
- *Joint input-output identification:* Treat both $u(t)$ and $y(t)$ as inputs of a system driven by $v(t)$ and noise. Then identify the characteristics of the feedback and plant using a joint model.

For PHIL applications our HUT is likely to be a closed-loop system such as a solar inverter, wind turbine, synchronous generator, or any other active electrical machine. For the case of a grid tied inverter, the model shown in figure 4.43 is particularly helpful, though the conclusion derived from it can apply to a variety of different HUTs. Given that we are simply interested in the dynamics of the output impedance of the HUT, this will be our plant $G_s(q^{-1})$. The current across this impedance will be the output of our plant, $y(t)$, while the voltage across the impedance will be $u(t)$. It is easy to see that the voltage across the impedance will be the difference between the voltage of the grid (or the PHIL amplifier in this case) and the voltage generated by the HUT. In our diagram, this will be the difference between $v(t)$ and $f(t)$. Finally, the rest of the inverter, both the control logic and the behaviour as a voltage source, can be represented as $F(q^{-1})$. This, as the goal of an inverter in a grid tied scenario, is essentially to control the flow of current through its POI by manipulating voltage generated by the switching of its transistors. With this insight,

we can now assess the feasibility of each of the presented approaches for identifying the output impedance.

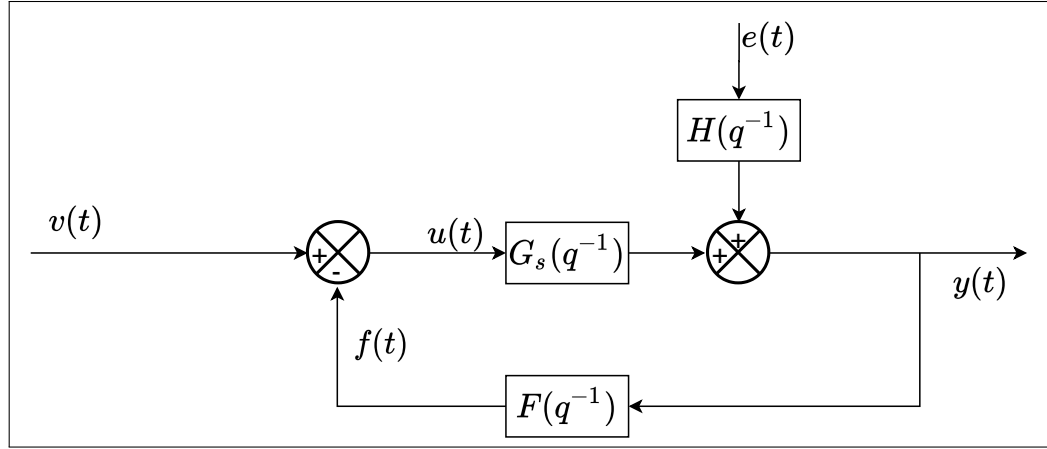


FIGURE 4.43. Identification scenario of an active HUT

Ideally, we would simply use the direct approach as it is the most straightforward way of identifying a closed-loop system. Simply measure $u(t)$ and $y(t)$ and fit a model accordingly. Nonetheless, this presents serious practical challenges, particularly, measuring $u(t)$. In most HUTs the output impedance is not readily accessible for measurements making the direct measurement of $u(t)$ impossible.

If the direct approach cannot be applied, then using the indirect approach may be the best option. The measurements for this method should be always available as only the amplifier voltage and current are required. However, in this case what will be identified will be the dynamics of the whole HUT, feedback included. Thus, to separate the dynamics of the HUT's output impedance needed for the DIM algorithm, prior knowledge of $F(q^{-1})$ is needed. This poses challenges as the control laws of the HUT may not be known or provided by the manufacturer. Obtaining them from the dynamics of the inverter, will present a complex challenge all on its own.

Finally the joint input-output approach is met with the same difficulty of the direct approach, the lack of a direct way of measuring $u(t)$. This approach is used when the dynamics of the whole system are of interest, not just $G_s(q^{-1})$. Thus, if $u(t)$ can be measured, it could be used to not only identify the output impedance, but also the control dynamics of the HUT.

Given this analysis, we should conclude that a straightforward way of identifying the output impedance, as presented in Chapter 3, of closed-loop system without any prior knowledge of it, is likely not possible, at least if $u(t)$ cannot be measured. Any algorithms seeking to continuously re-identify Z_{HUT} of a closed loop HUT during the simulation will face the challenges described above.

As such, the need for continuous re-identification of $G_s(q^{-1})$ should be carefully examined as it poses the biggest challenge for a successful PHIL simulation using the DIM algorithm. Given that most changes in the behaviour of an active HUT are due to changes in its control laws $F(q^{-1})$, instead of its output impedance $G_s(q^{-1})$, for some experiment it can be entirely possible to obtain accurate results by identifying Z_{HUT} while the HUT is powered off. During this time, and depending on HUT topology, the identification procedure can be carried out effectively as described in Chapter 3, as $F(q^{-1})$ will essentially be 0 and the system will be in an open loop condition. Then, once a model of the output impedance has been obtained, the PHIL simulation can commence. This approach will produce accurate results, as long as the physical parameters of Z_{HUT} don't change significantly during the experiment, or if there isn't an significant topological change when the HUT is turned on. Whether this strategy is applicable or not to an experiment, is something that must be determined case by case.

Despite the above, identification of the HUT's output impedance while powered off, can provide relatively simple way for achieving stable and highly accurate PHIL experiments involving active HUTs, whenever the conditions are right for its application.

To back the recommendations made above, experimental results are presented in the next Chapter. We will show how online identification of a passive HUT is not a problem, but that as soon as the HUT is a close-loop system, the estimates for Z_{HUT} become biased and accuracy suffers greatly. Then we test the same identification routine to identify the output impedance of the active HUT while it is powered off and keep this estimate to use it during the simulation. We show how this approach delivers the same degree of stability and accuracy as the passive HUT case with the drawback that the impedance cannot be re-identified while the simulation is running.

4.8. Chapter conclusion

In this Chapter we have presented the structure of the PHIL platform as well as its implementation in Simulink, following the design philosophy needed for a later implementation in a real time target. We showed how the model is divided into several subsystems, each with a very distinct task, and how they all relate with one another.

The subsystem *SM_computations* implements the DIM interface algorithm. While most of it is fairly straight forward, the implementation of a variable impedance, capable of representing any SISO transfer function represented a particular challenge.

The identification routine was implemented in the subsystem *SS_ID*. In this subsystem, both the spectral analysis and the parametric fitting were programmed using the custom Matlab function block. This allowed us to translate the Matlab scripts used in previous sections to blocks fit for their use in Simulink.

Along with the software implementation, the results of the software testing of the platform were delivered as well. Here we corroborate that for the case of passive HUTs, the platform performs well. Nonetheless, we discover that for active HUTs spectral analysis fails to deliver good results. An explanation for these results is then given, and a deeper analysis of the identification of closed-loop systems in the environment of a PHIL platform is given. Through this analysis, possible solutions and their challenges are

theorised and proposed. Finally, a workaround is presented that allows us to use the existing identification routine to achieve successful experiments. This is to simply identify the HUT while it is powered off, as during that time, it will behave as an open loop system. This solution should suffice for the needs of the majority of PHIL experiments though it is not without drawbacks as we sacrifice the ability to re-identify the output impedance of the HUT during the simulation.

5. HARDWARE IMPLEMENTATION & EXPERIMENTAL RESULTS

5.1. Introduction

As explained in chapter 4 the software for the real time target will be based on the simulation presented in that chapter. RT-LAB allows us to take a working Simulink model and add a few specialised blocks to transform it into a program that can be compiled and run in real time. The first part of this chapter will focus on the general programming guidelines to follow and things to avoid when programming for real time with Simulink and RT-LAB. Then changes needed to get the simulation presented in chapter 4 to run in the real time target will be revised and explained. Finally we will cover the physical part of the PHIL platform and how it interfaces with the simulation.

5.2. Guidelines for programming for real time with Matlab Simulink and RT-LAB

In order to ensure that a Simulink can be properly compiled into code that can run in real-time on the OPAL-RT platform, a few guidelines and considerations must be followed. Here we present several that one must keep in mind to avoid compilation or run-time errors, later in the development process.

5.2.1. Discrete solvers and algebraic loops

To allow for real time execution, several measures have to be taken to ensure the final result can run outside Simulink. First and foremost, the simulation must be discrete, with a solver with a fixed time step. Continuous time simulation or even discrete time simulations with variable time steps solvers cannot be compiled into c code and thus those options must not be selected.

A far less obvious consideration, are algebraic loops. Algebraic loops often occur in feedback paths, very common when programming in Simulink, when the input for a certain calculation requires its output in the first place to be already available, an example

can be seen in figure 5.1 . While Simulink can normally break these, programs meant to be run in real time can't, so these will result in compilation errors.

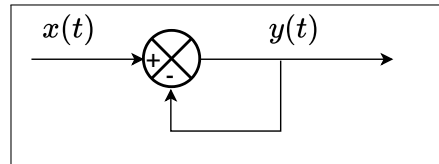


FIGURE 5.1. Algebraic loop example

Thankfully these loops have a rather simple fix, inserting a memory or delay block in the feedback path will break the loop as the simulation can use the previous state of the output to calculate the input.

5.2.1.1. Subsystem communication

As previously explained, RT-LAB separates each subsystem and assigns it to a different core of the target. For this to work properly the communication between subsystems must be handled in specific ways. The first consideration is the addition of OpComm block. These blocks serve three main functions:

1. They tell RT-LAB how subsystems must be communicated, this is crucial as this connections between cores become hardware communication links when the deployed in the target
2. They provide RT lab information about the types of variables communicated between subsystems
3. When inserted in the console subsystem, they allow for the monitoring of signals from the simulation via acquisition groups.

Subsystem's outputs must also be carefully managed in order to achieve an efficient execution in real time. During execution, all subsystems will try run their code at the start of each timestep, however if the input of one subsystem depends on the output of another, this subsystem must wait until the output is available. This may lead to two subsystems instead of executing one after the other, or even enter into a dead lock, instead of running in

parallel defeating the purpose of assigning them to different cores. This can be prevented by placing memory, delays or integrator blocks before the outputs of the subsystems, this way, the results of the previous timestep will be available at the start of the current timestep to the receiving subsystem, allowing both to run in parallel. Examples of this can be seen in figures 5.2, 5.3, 5.4, taken from OPAL-RT ([Wei, 2017](#)).

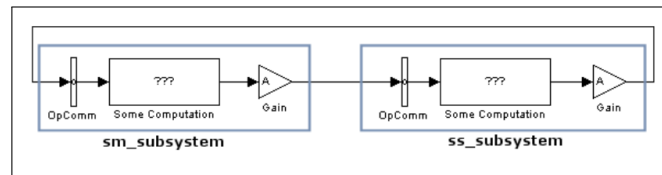


FIGURE 5.2. Subsystems at deadlock

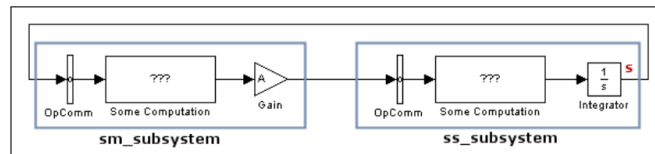


FIGURE 5.3. The subsystems executing in series

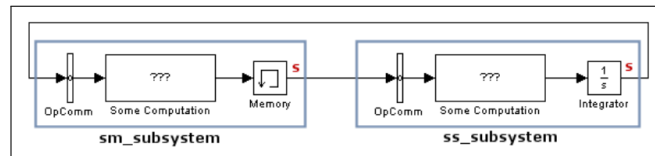


FIGURE 5.4. Subsystems executing in parallel

5.2.2. Using code compatible with code generation

All code used in the Simulink model will be compiled into a C-file via Matlab coder and loaded on to the real time target. However there is a great number of Matlab commands which are not compatible with code generation and thus should not be used in any part of the model or in scripts that the model will run at any given time. Another concern is variable size and the corresponding memory allocation. When compiling code to be deployed in embedded systems the compiler must know or be able to predict the size of all variables

in the program, signals with variable size must be avoided or at least, their maximum size must be specified so that the memory allocation can occur. In fact, unpredictability of output size is one of the primary reasons commands are not supported for code generation. This can be a concern if custom functions are used in the program as Matlab must be able to their output size. Further reading about Matlab functions compatible with code generation can be found in ([Mathworks, 2020](#))

5.3. Changes to subsystems

Taking into consideration the guidelines presented above, we will proceed to make changes to the model's subsystems to allow for it to be compiled and loaded into the target.

The main changes to SM_computations were that the HUT is now replaced by the I/O module. This module handles the analog output that controls the voltage amplifier, as well as the analog inputs coming from the sensors. To do this, it uses specialised blocks provided by OPAL-RT which configure the FPGA that handles the I/O. Three other signals are sent to the analog outputs: the sensed voltage of the HUT, the current of the simulation, and the current of the HUT. These 4 analog signals are sent to BNC outputs so they can be easily viewed with an oscilloscope.

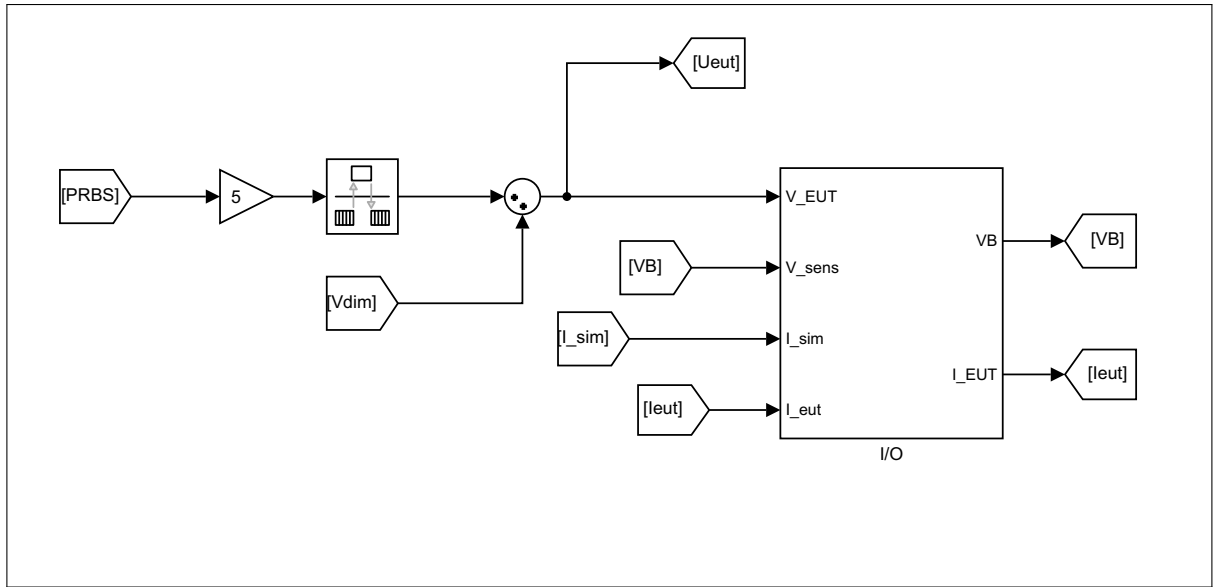


FIGURE 5.5. Block handling I/O

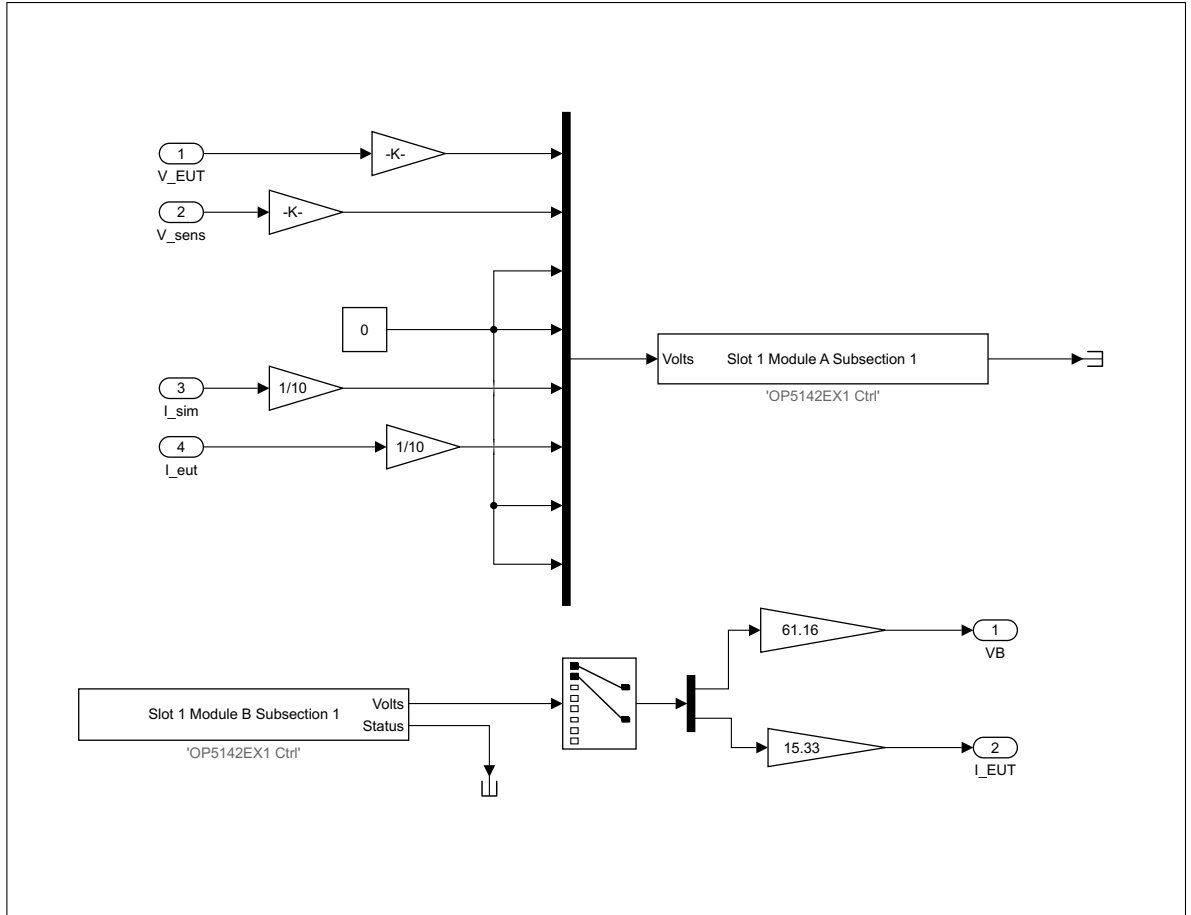


FIGURE 5.6. Detail of I/O block

To allow for better balance of the computational load and to take more advantage from the parallelisation, the sampler that forms the frame for the identification subsystem is moved from SM_computations to SS_measurements subsystem as this system showed less cpu utilisation during testing.

All subsystems were fitted with Opcomm block, to handle communication with other subsystems. Monitoring block were added as well, to review the performance of the simulation and check that all subsystems were completing their calculation in their designated timesteps. Opwrite block were added to SM_computation and SS_measurements to store the measurements into files for later processing. Finally, delays and memory block were

put at the outputs of the subsystems to avoid the execution problems laid out in section 5.2.1.1.

5.4. Physical components of PHIL platform

5.4.1. OPAL-RT target: OP5600

The real time simulation target is an OP5600 from OPAL-RT. These real time simulation targets are specialised computers designed execute the code with precise timing and handle the incoming and outgoing signals required to run a PHIL simulation.

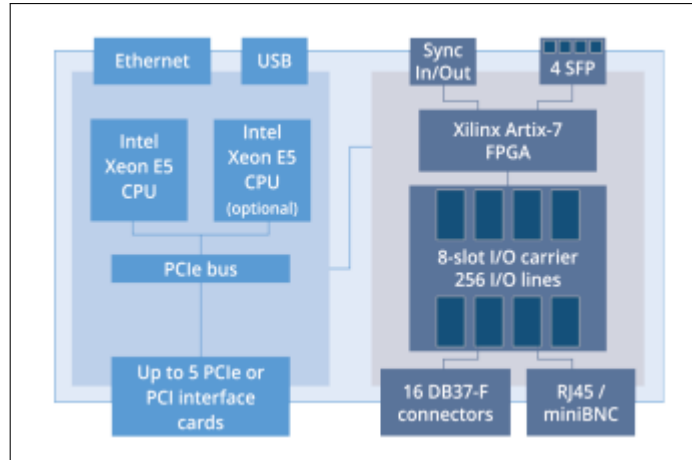


FIGURE 5.7. Real time simulator internal architecture.

The device has two main parts: one dedicated to computation and one dedicated to I/O. The simulator used during testing has two CPUs of six cores each, for a total of 12 available cores. Each core runs at a speed of 3Ghz giving the simulator ample power to run code in real time. The I/O part correspond to up to 5 FPGAs which can be fitted into the system. Our simulator had 1 analog I/O card and 1 digital I/O card. Providing in total: 16 analog outputs, 16 analog inputs and 32 digital I/O (OPAL-RT, 2020a) . The device also has 5 PCIe expansions slots, enabling the addition of many available models. The simulator used was installed with a CANBUS communication card and an Ethernet port expansion card.

5.4.2. Voltage amplifier

The voltage amplifier is the physical component that forms the interface between the HUT and the simulation. Because of this, it is crucial that the device is able to respond to commands given by the simulation with a very short delay. It must also be able to provide or sink all the power required by the HUT. The amplifier used in the PHIL platform is the PA-3X7000-AC-DC-4Q-400V-54A-4G from puissance plus. The device is a 3-phase, 4-quadrant linear voltage amplifier capable of delivering up to 7kW per phase. While sinking power the rating is reduced as shown in figure . Given that the power used during the experimental tests will not surpass 4kVA, the amplifier is more that capable of supplying and sinking all the necessary power. A chart provided by the manufacturer of the operational limits of the device is shown in figure 5.8.

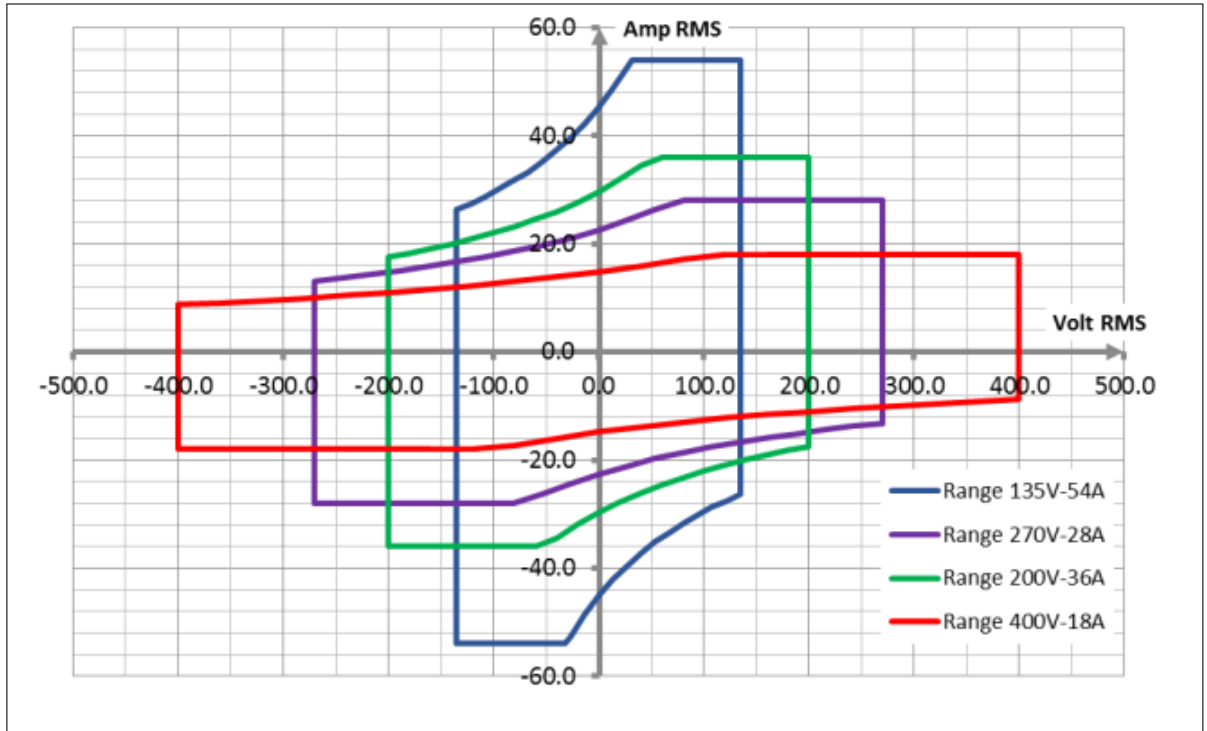


FIGURE 5.8. Operational limits of Amplifier

The device is also very fast with the manufacturer guaranteeing rise and fall times of less than $7\mu s$ and with a voltage regulation bandwidth of 70kHz. With these characteristics, we can foresee that the amplifier dynamics will have very little impact on the performance of the PHIL platform, as the slower dynamics of the rest of the simulation will dominate, given that the electrical simulation timestep is $100\mu s$. Figures 5.9 and 5.10 show the manufacturer's provided data on bandwidth and step response. It is worth noting that our device was not equipped to function as a current source so the current regulation curve, in red, can be ignored. Further information about the device can be seen in its datasheet ([Puissance Plus, 2020](#)).

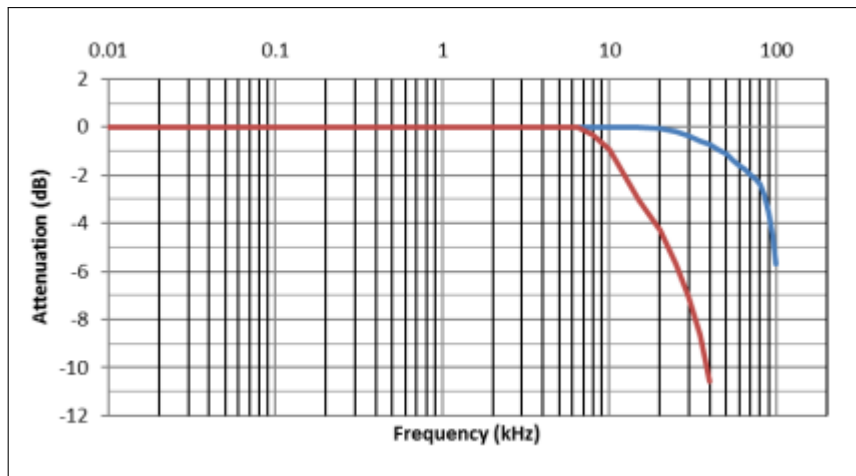


FIGURE 5.9. Amplifier bandwidth

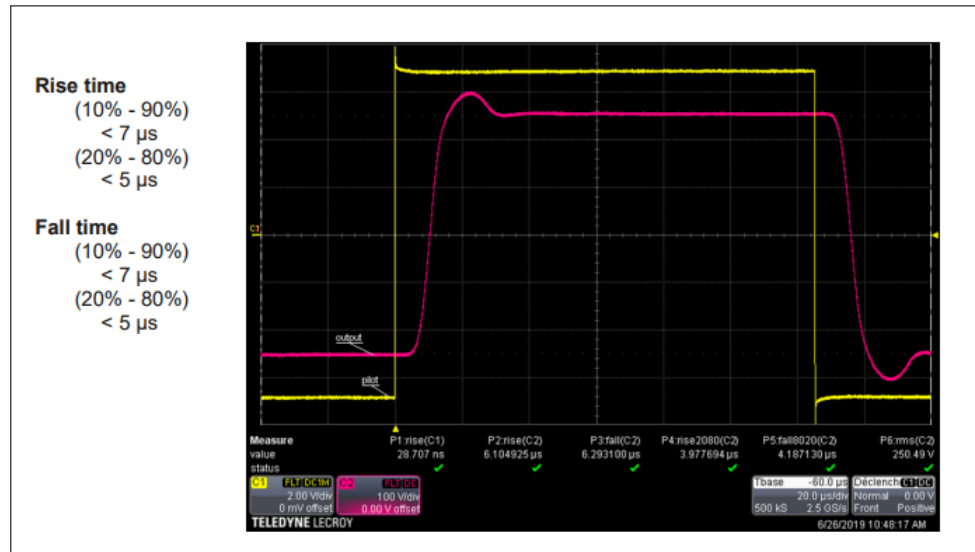


FIGURE 5.10. Amplifier step response

5.4.3. Delta electronica DC source

A high power DC source was an accessory part of the platform. The specific model was the SM 1500-CP-30 from Delta electronika, this bi-directional power supply is capable of providing between 0-1500V DC and delivering or sinking up to 15kW of power. The device can be controlled via its front panel as well as through telnet commands. The device may also be controlled via an analog input, potentially turning it into an amplifier however the device dynamic is far too slow for that. The main goal of this device is to power the solar inverter during testing. Further information about the SM 1500-CP-30 can be found in ([Delta elektronika B.V., 2020](#))



FIGURE 5.11. Delta electronika DC source

5.4.4. HUTs

Two HUTs were used in testing of the PHIL platform. The first HUT was an RL filter made out of a large resistance cage and in series with an inductor. Both components provided a large range of possible values with the inductor having 8 terminals allowing for 7 different inductance values, ranging from 2 to 18 mH . In testing, the value of 9 mH was used. The resistance cage also allowed for several different connections, however given its low overall resistance, all possible connections were put in series, adding up to 8.8 Ω .

The second HUT used was a residential solar inverter from Chinese manufacturer Huawei (SUN2000L-3KTL). The grid tied, single phase inverter had a rated AC power output of 3kW and a maximum apparent power of 3.3 kVA. On the DC side, the device could be connected to up to 4.5 kWp of solar panels split in two arrays and operating between 160VDC and 480VDC. A connection to a specific type of battery was also available. This inverter is of transformerless design, meaning there is no transformer isolating the inverter from the grid, only an output filter. A systems level diagram of the inverter can be seen in figure 5.12. However, the manufacturer does not provide information about the filter topology. In fact, this will likely be the case for most HUTs tested in a PHIL platform so it is imperative that our identification routine is able to identify the device's impedance. Further information about the device can be found in its datasheet ([Huawei, 2016](#)) and in annex A.1. The topology of the device is very interesting as we can see that the output filter is after the isolation relays. This quality will prove useful during testing.

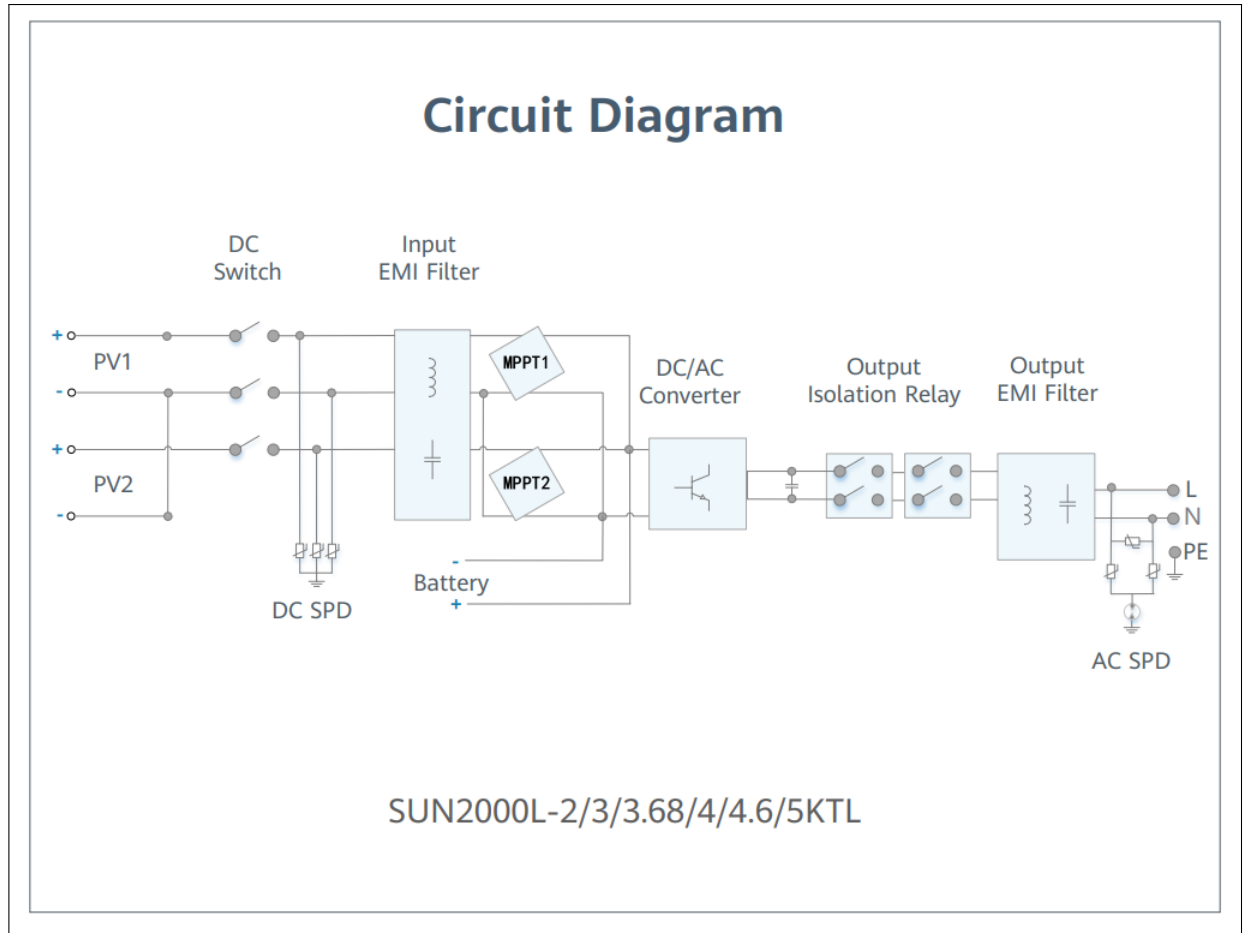


FIGURE 5.12. Huawei inverter topology

5.5. Testing: Passive load

This test replicated the conditions described in section where we simulated the PHIL platform with a an RL filter as load. In this instance, the HUT is now a real RL filter made out of a large resistance cage and a high power inductance. While the inductor was labeled 9 mH upon measuring with a universal bridge it measured 9.6 mH . The resistance measured $8.85\ \Omega$. The voltage amplifier seen in section 5.4.2 is used to make the interface between the real time simulator and the HUT. The voltage level set to was to 230V RMS

and a 5Ω impedance was placed as software impedance. The layout can be see figure 5.13. The parameters selected for the simulation can be seen in table 5.1.

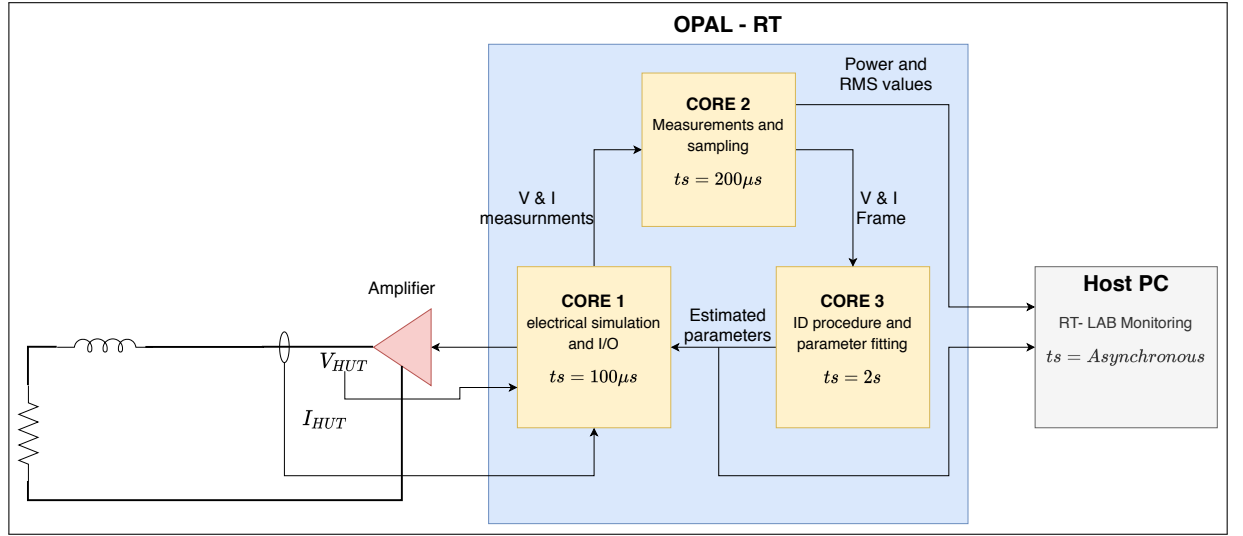


FIGURE 5.13. Experiment setup with passive load

TABLE 5.1. Experiment parameters: passive load

Parameter	Value
N	2000
P	5
Window size (γ)	250
SM_computations timestep	$100\mu s$
SS_RMS timestep	$200\mu s$
SS_ID timestep	$2s$
Software impedance (Z_s)	5Ω

Results were satisfactory, as expected, given that the load was passive and the results from the spectral analysis should be reliable. Given that the topology of the HUT is known, we can simulate the naturally coupled circuit and obtain a reference as we did with the

simulation results. We observe that during the start of the simulation fit is poor, however once Z_{HUT} is identified, the simulation becomes highly accurate.

Results for the voltage can be seen figures 5.14, 5.15 and 5.16. The results obtained are consistent with the simulation with the voltage of both hardware and simulation reaming close to the reference. The relative error ranges for the simulation voltage is on average 2 % while the hardware voltage has a deviation of 1 % with respect to the reference. The results for the current are similar and can be seen in figures 5.17 and 5.18 5.19 with the relative error of both the simulation and hardware of 2 %. Finally, active and reactive power show the same dynamic as the voltage and current. Figures 5.20, 5.21 and 5.22 shows that simulation power is dead on the reference while the hardware is only 1 % off. The results are similarly good for reactive power as seen in figures 5.23, 5.24 and 5.25 , where simulation reactive power is almost at the reference and while the hardware presents a maximum error of 2 %.

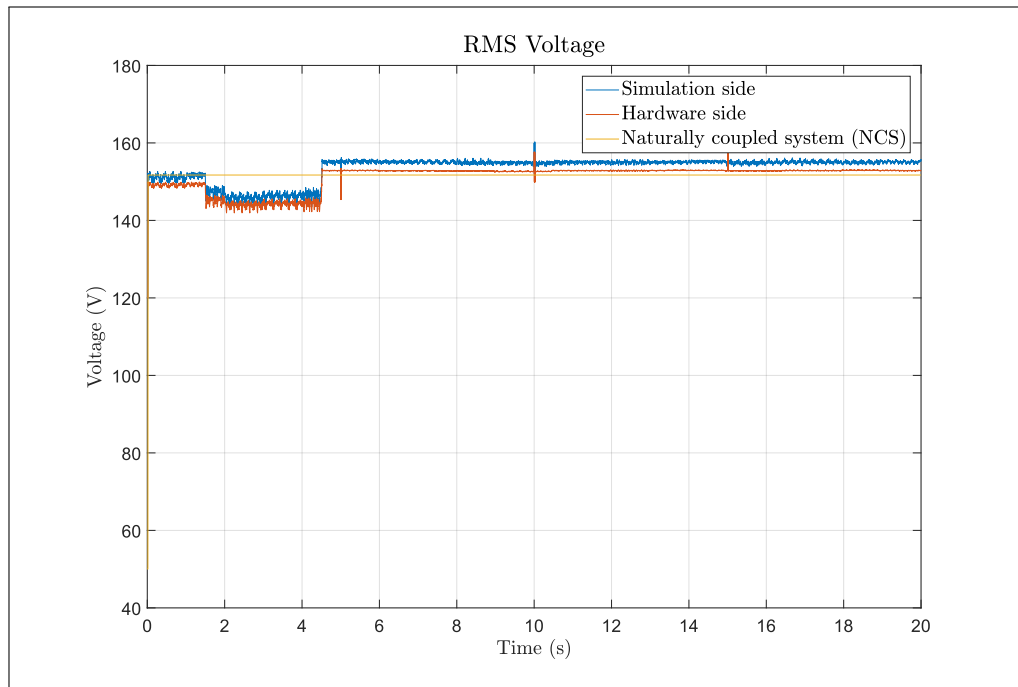


FIGURE 5.14. Experiment with passive load: RMS Voltage.

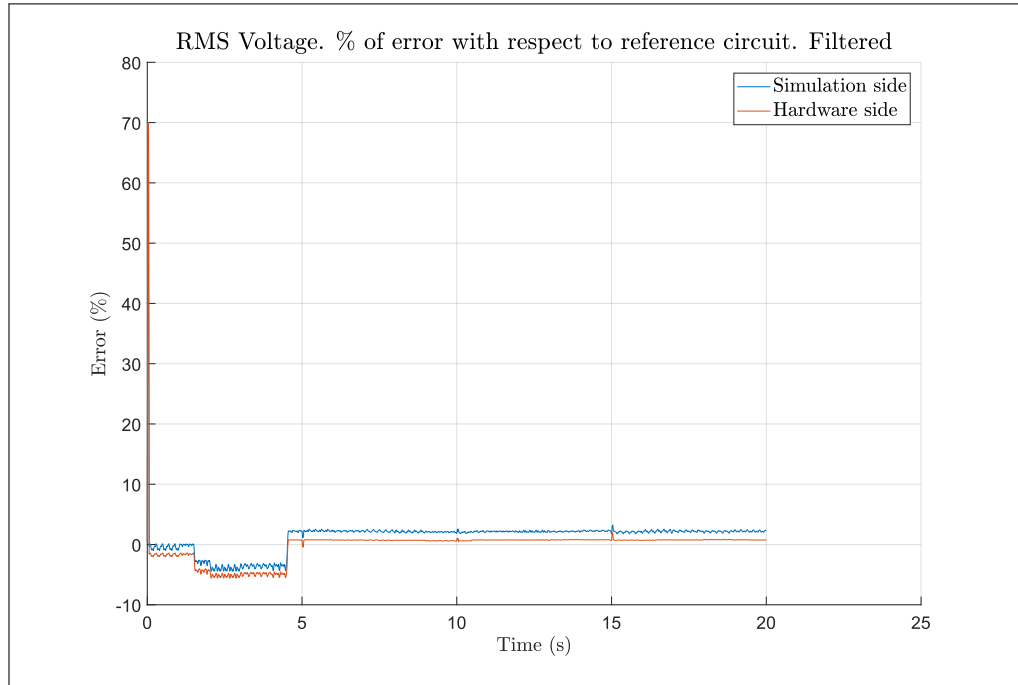


FIGURE 5.15. Experiment with passive load: % of error for RMS voltage.

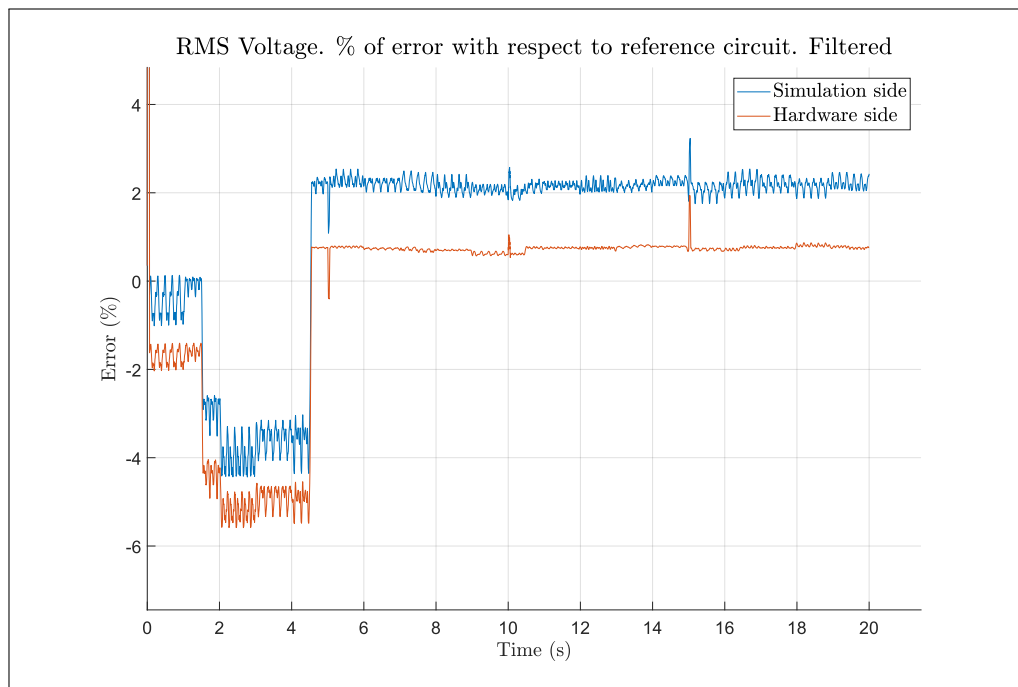


FIGURE 5.16. Experiment with passive load: % of error for RMS voltage. Zoom

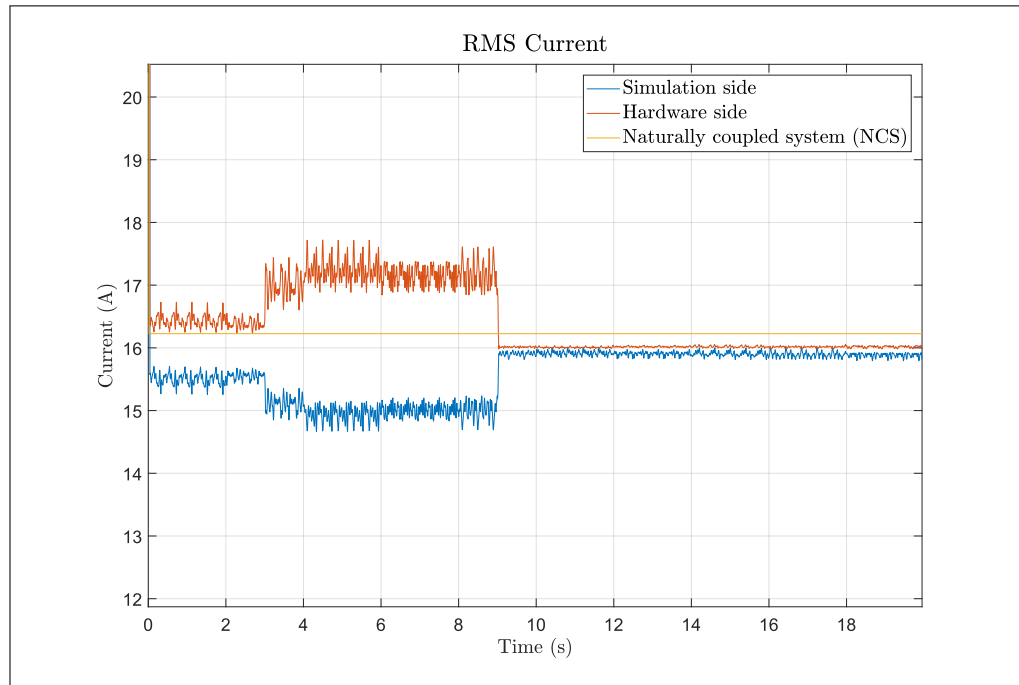


FIGURE 5.17. Experiment with passive load: RMS current.

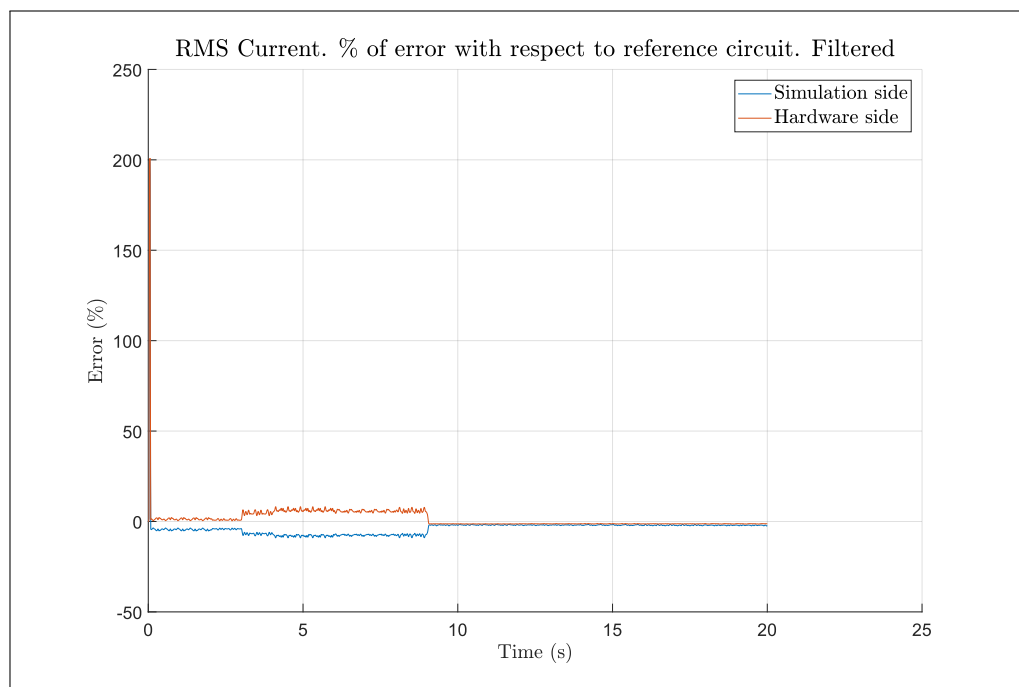


FIGURE 5.18. Experiment with passive load: % of error for RMS current.

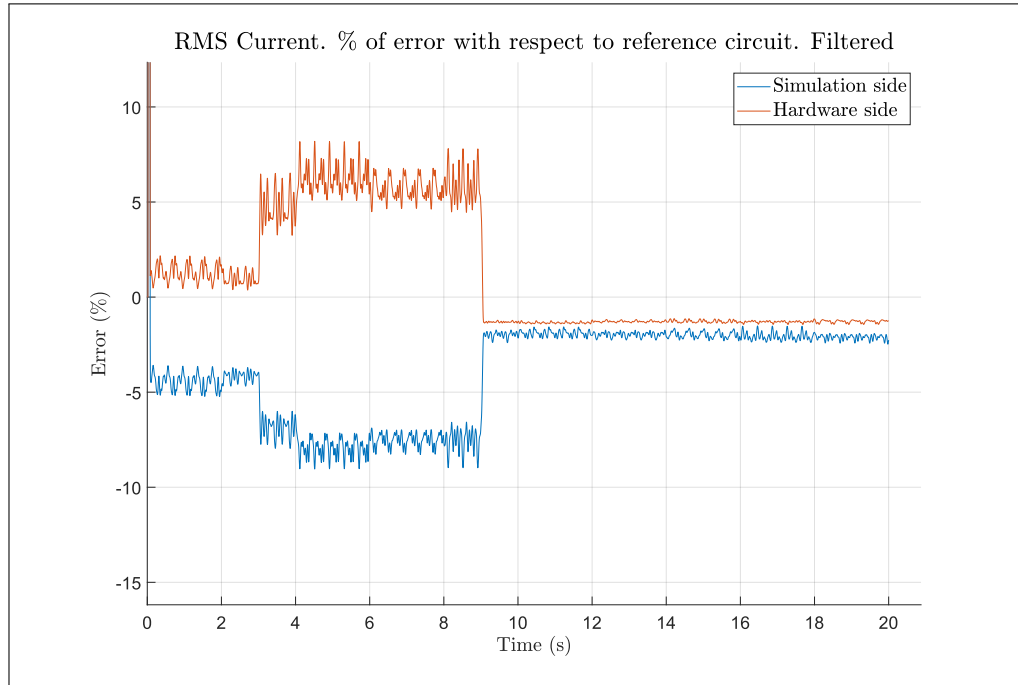


FIGURE 5.19. Experiment with passive load: % of error for RMS current. Zoom

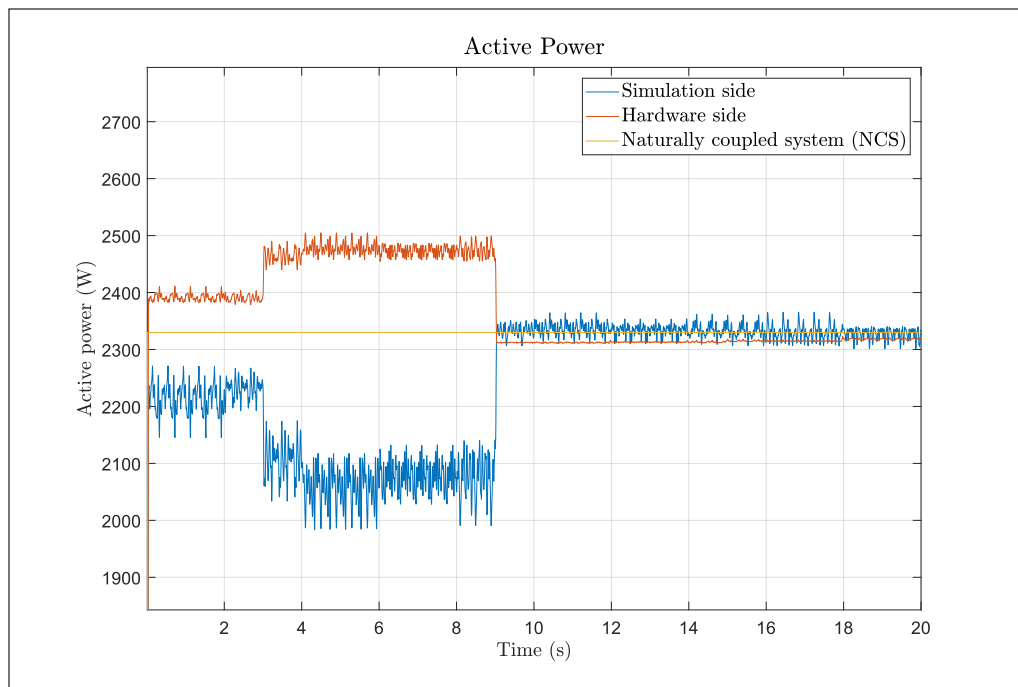


FIGURE 5.20. Experiment with passive load: Active power.

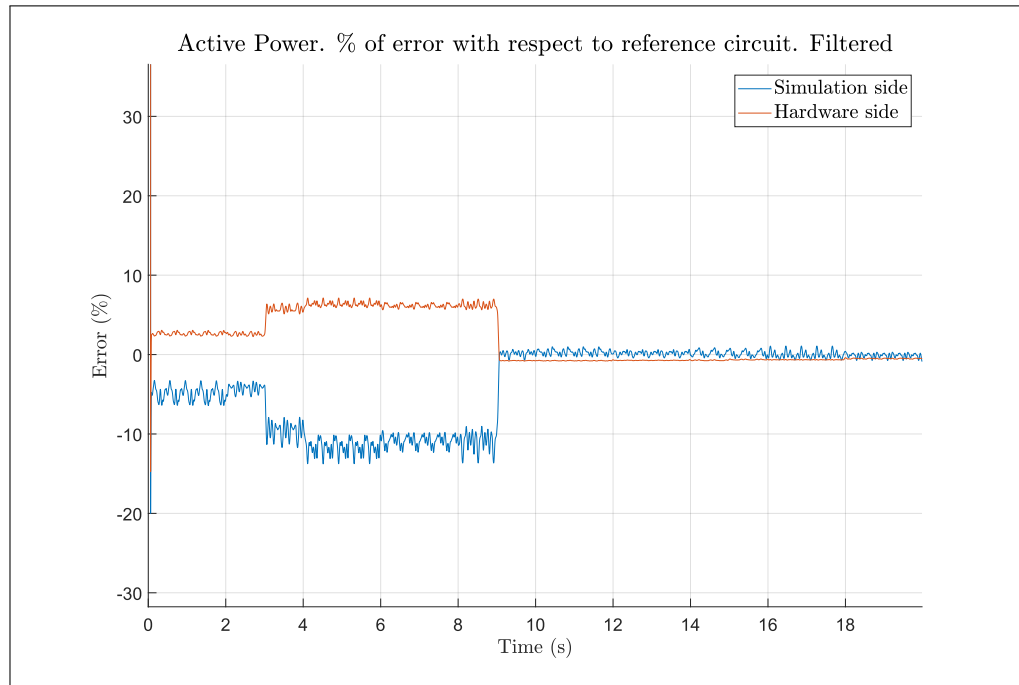


FIGURE 5.21. Experiment with passive load: Active power. % of error for active power.

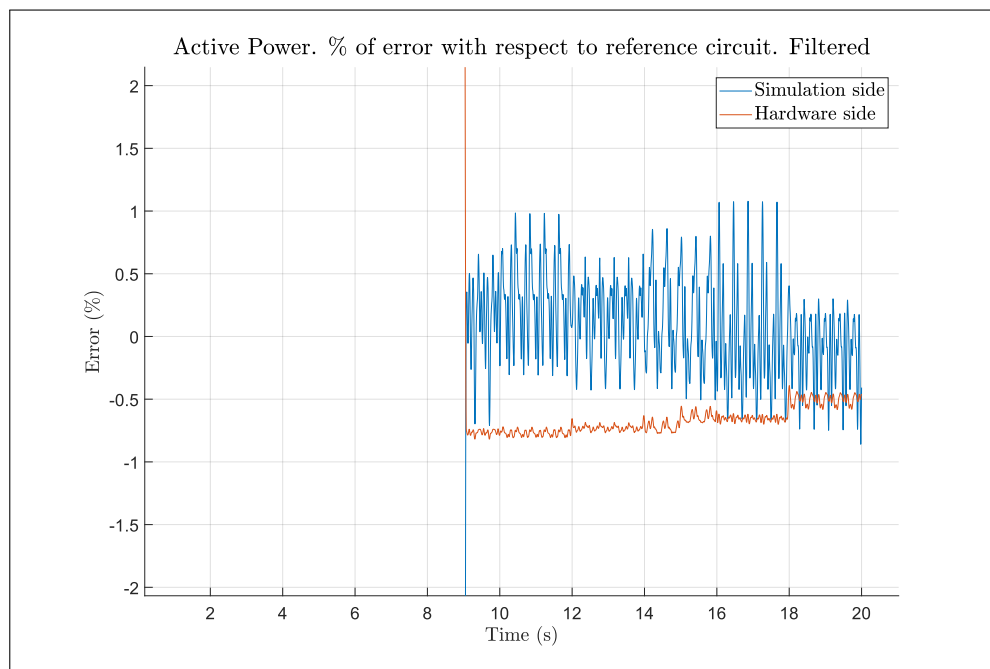


FIGURE 5.22. Experiment with passive load: % of error for active power. Zoom.

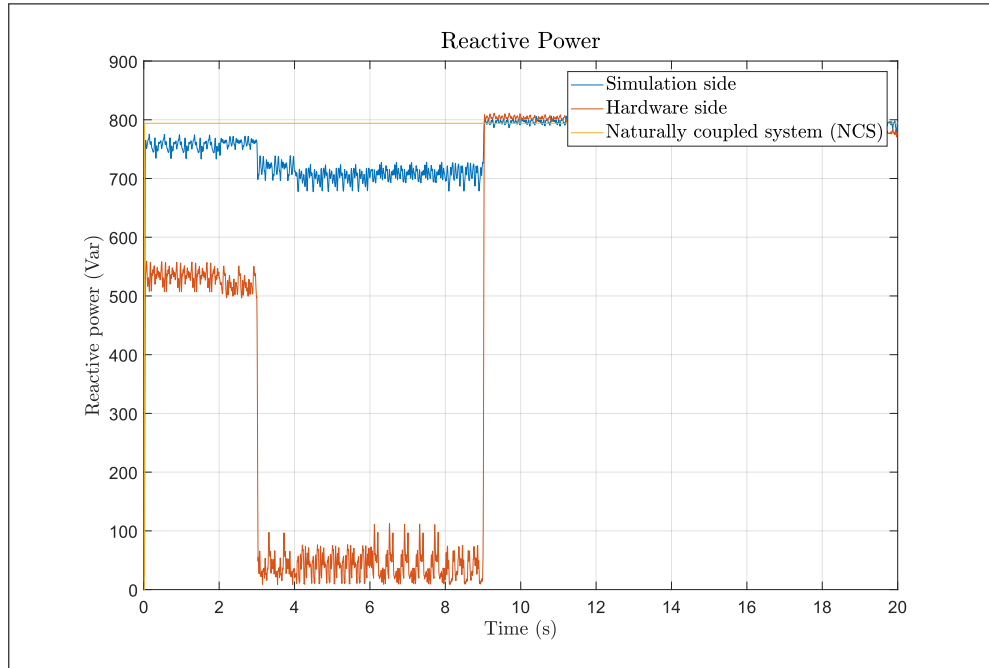


FIGURE 5.23. Experiment with passive load: Reactive power.

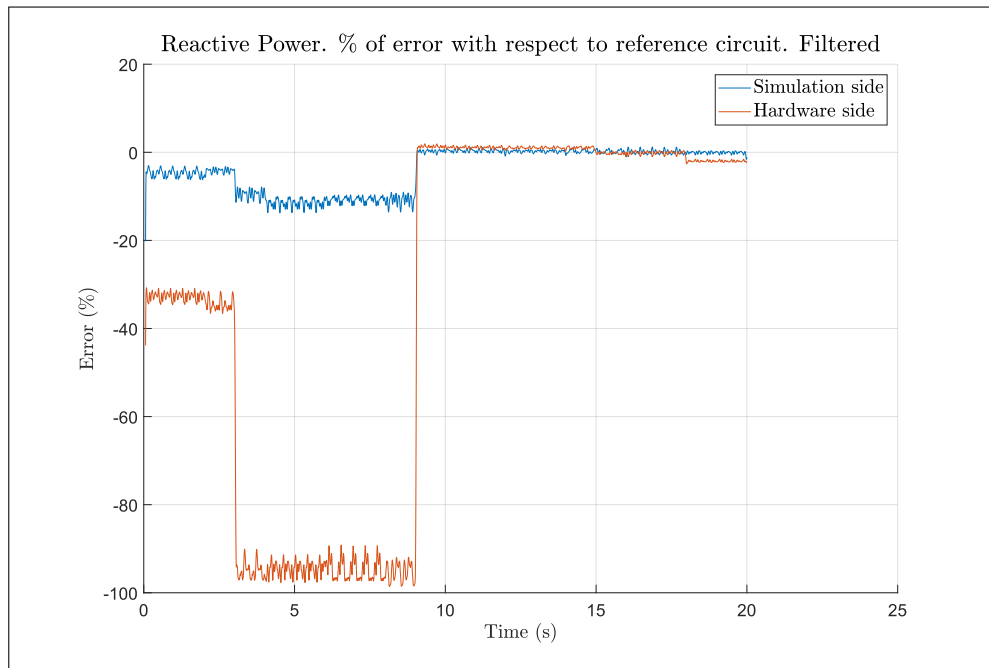


FIGURE 5.24. Experiment with passive load: % of error for reactive power.

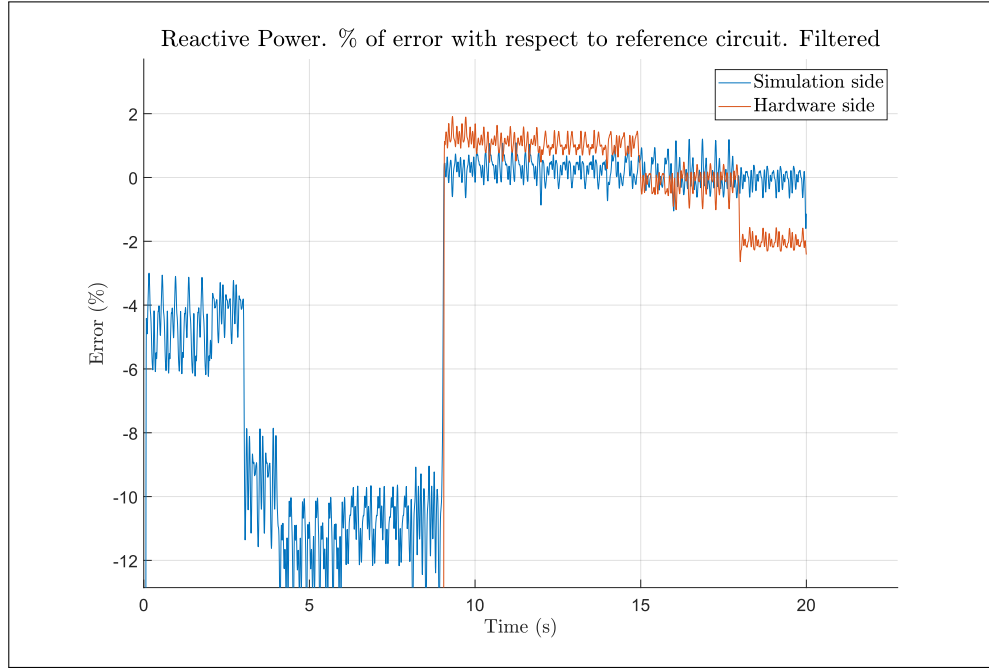


FIGURE 5.25. Experiment with passive load: % of error for reactive power. Zoom.

5.6. Testing: Solar inverter

Following the test with a passive load, we proceed to test the platform with a solar inverter as HUT powered by a DC source. The inverter used is the Huawei SUN2000L-3KTL described in section 5.4.4 while the DC source is the SM 1500-CP-30 from Delta Electronika shown in section 5.4.3. True to our goal of creating a system that can adapt to any HUT, no changes were made to the code of the systems from the previous test. Parameters such as timestep and number of samples were also maintained. Figure 5.26 shows the setup for this experiment. The only change is on the impedance on the software side as it was given values more representative of network connection. Table 5.2 shows the parameters used for this test.

Results from this test are initially disappointing showing the same problems the simulation testing revealed: the spectral analysis provides biased models for the filter impedance due to the effect of feedback from the inverter control. This can be seen in the

figures 5.29 and 5.30 which show how the active and reactive power behaved during the test. While no reference is available for this test as we naturally coupled system cannot be simulated, because of the lack of a detailed model of the inverter, we can be sure that the results of the test are inaccurate as the variables of the software and hardware sides do not converge.

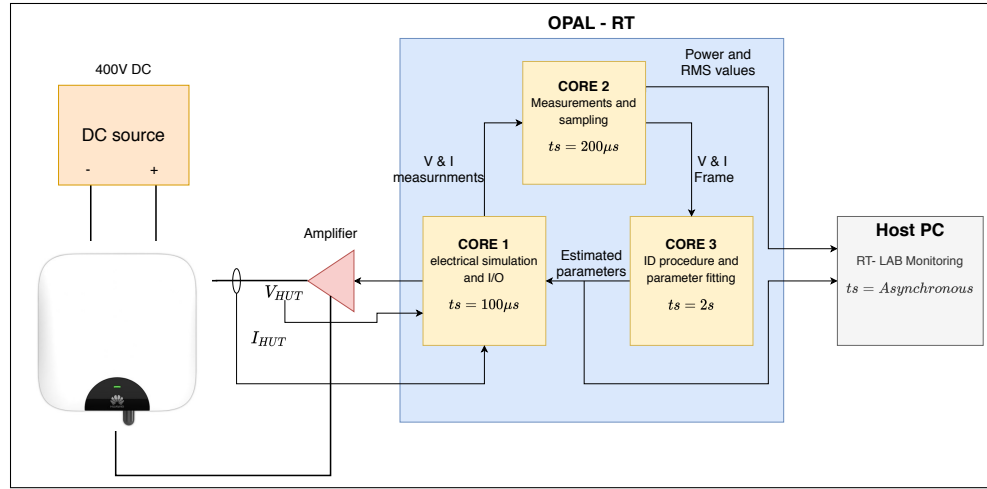


FIGURE 5.26. Experiment setup with solar inverter

TABLE 5.2. Experiment parameters: inverter load

Parameter	Value
N	2000
P	5
Window size (γ)	250
SM_computations timestep	$100\mu s$
SS_RMS timestep	$200\mu s$
SS_ID timestep	$2s$
Software impedance (Z_s)	$0,059\Omega + 10\mu H$

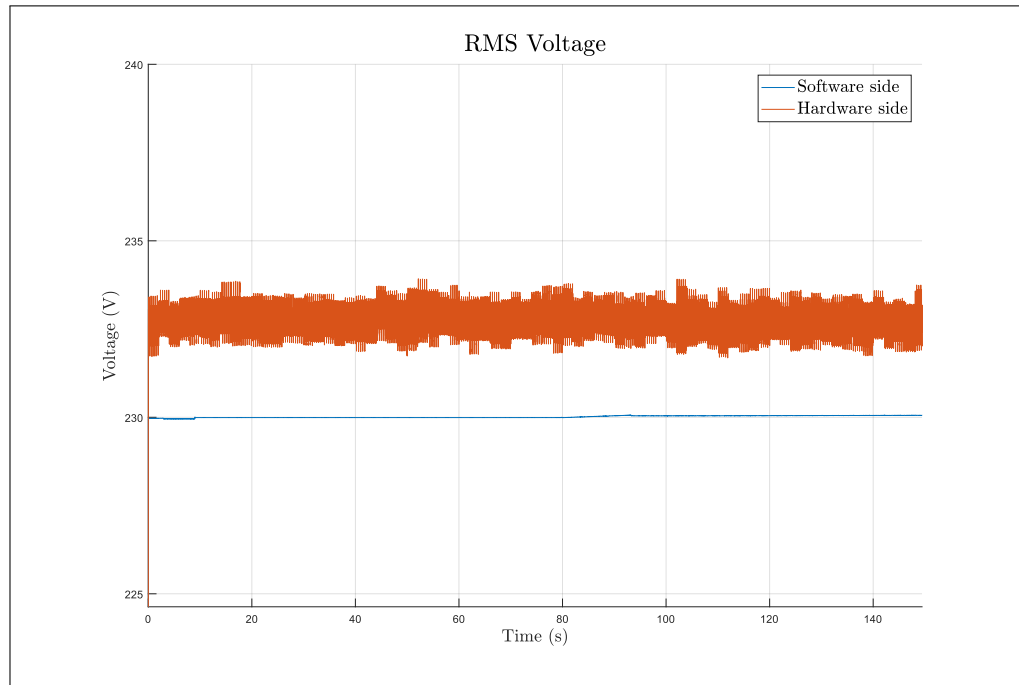


FIGURE 5.27. RMS voltage during test

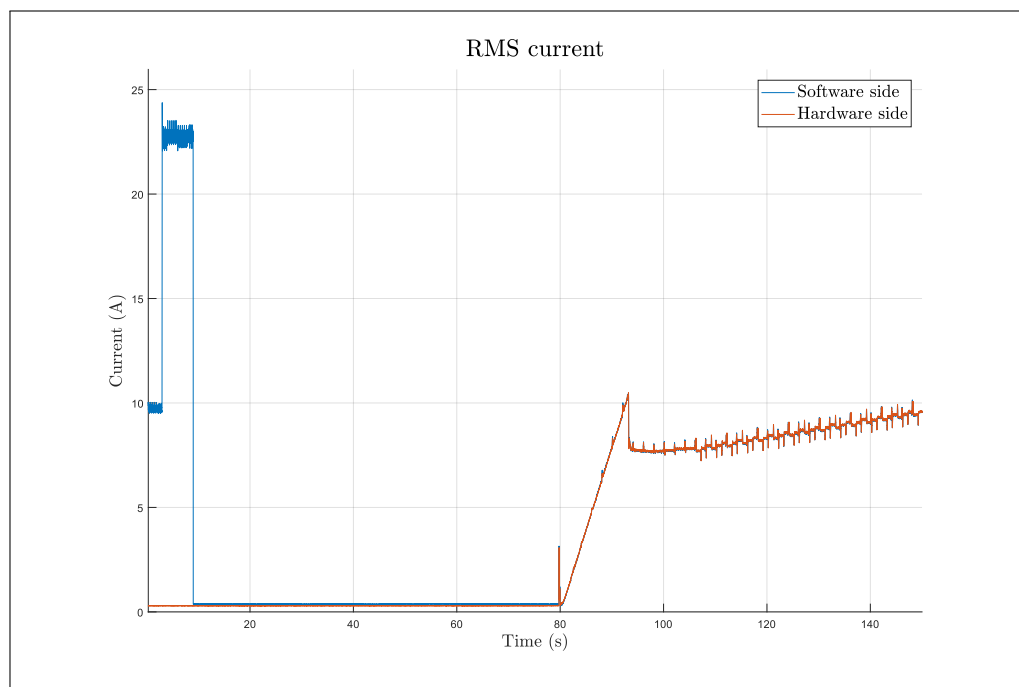


FIGURE 5.28. RMS current during test

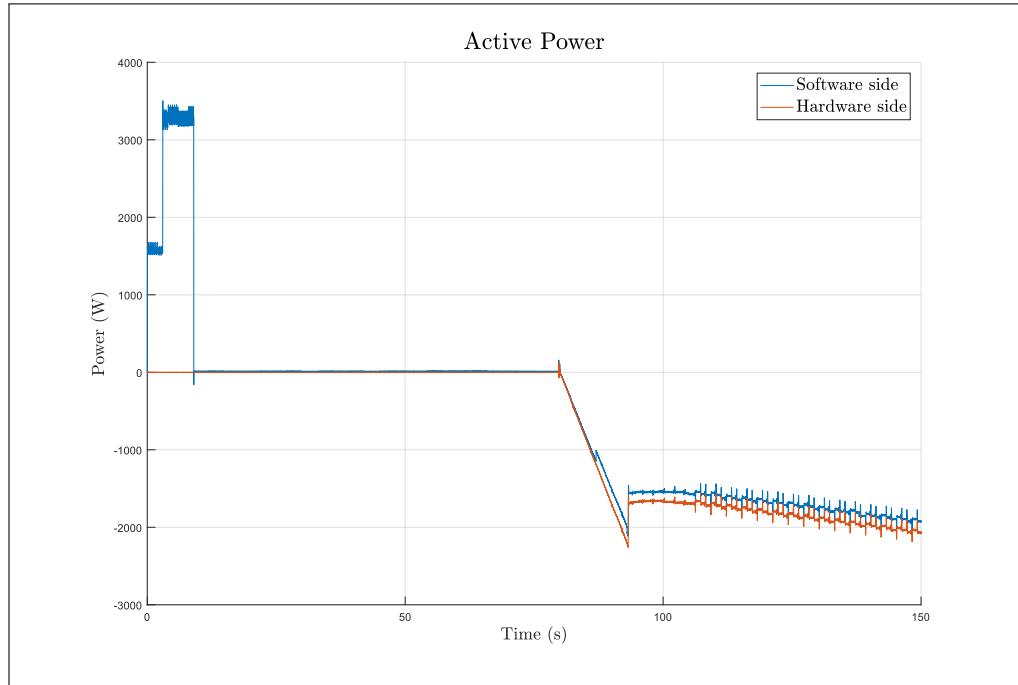


FIGURE 5.29. Active power during test

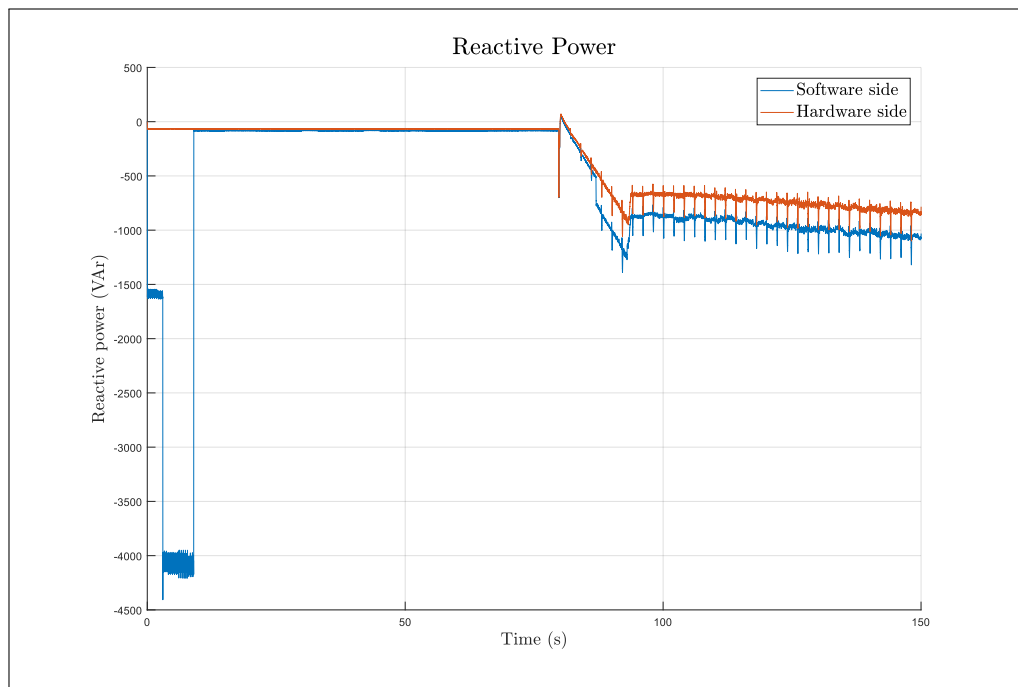


FIGURE 5.30. Reactive power during test

However, closer inspections of the results reveals a possible solution. The before starting to supply power the inverter goes through a connection process taking approximately 80 seconds. During this process the inverter checks, the voltage levels in the connection point, checks the grounding of the connection and connects to the internet. During this process the connection relays of the device are open. Going back to the inverter topology in figure 5.12, we see that the filter of the device is past the connection relays meaning that in fact, it is always connected to the network. Now, looking at the curves for active and reactive power we notice that the software and hardware variables are matched during the connection process and only after the inverter start injecting power into the simulated grid, the simulation becomes inaccurate. In fact the simulations becomes inaccurate six seconds after the device start supplying energy, before that software and hardware variables are nearly identical. This happens because six seconds after the device start feeding the grid, the parameters for Z_{DIM} are updated with the first re identification performed while the control of the inverter had an impact on the filter. We can confirm this by looking at the current waveform before and after this change, as seen in figures 5.33, 5.34 and 5.35. We notice that after Z_{DIM} is updated with estimates obtained while the device was on, the current shows significant phase error. This gives us an idea: identify the filter before the inverter start providing power and then keep that estimate for the remainder of the simulation.

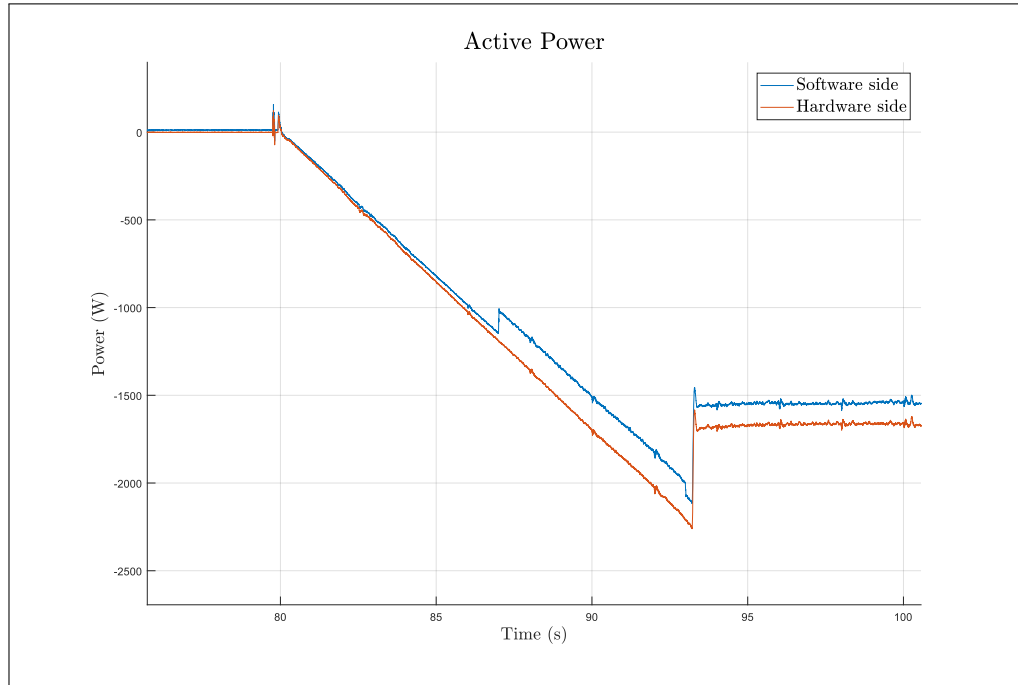


FIGURE 5.31. Active power during test. Zoom

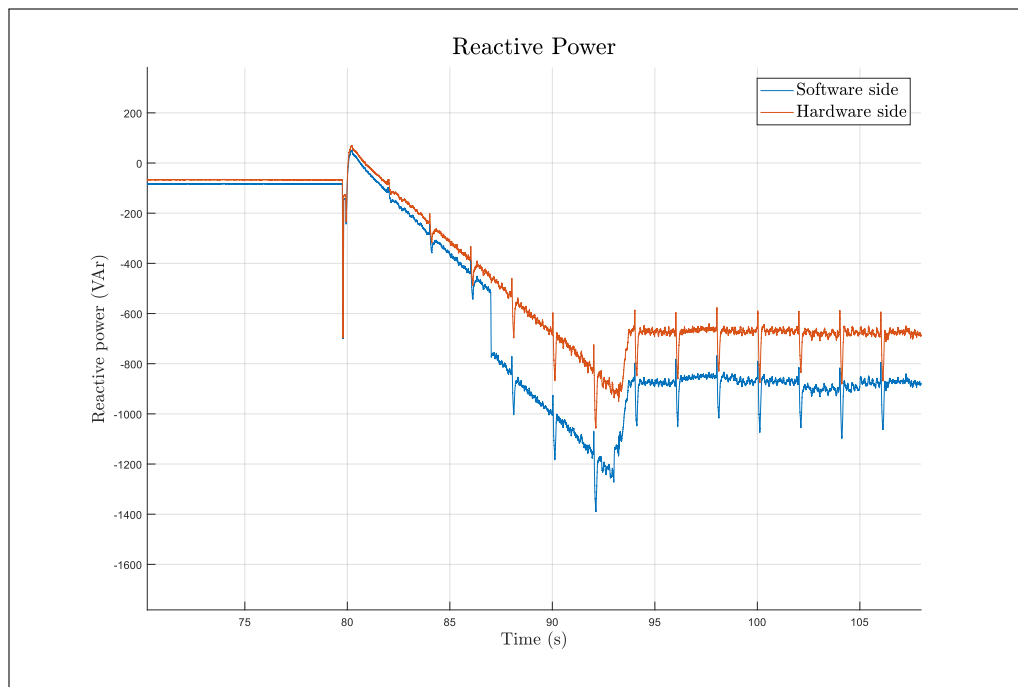


FIGURE 5.32. Reactive power during test. Zoom

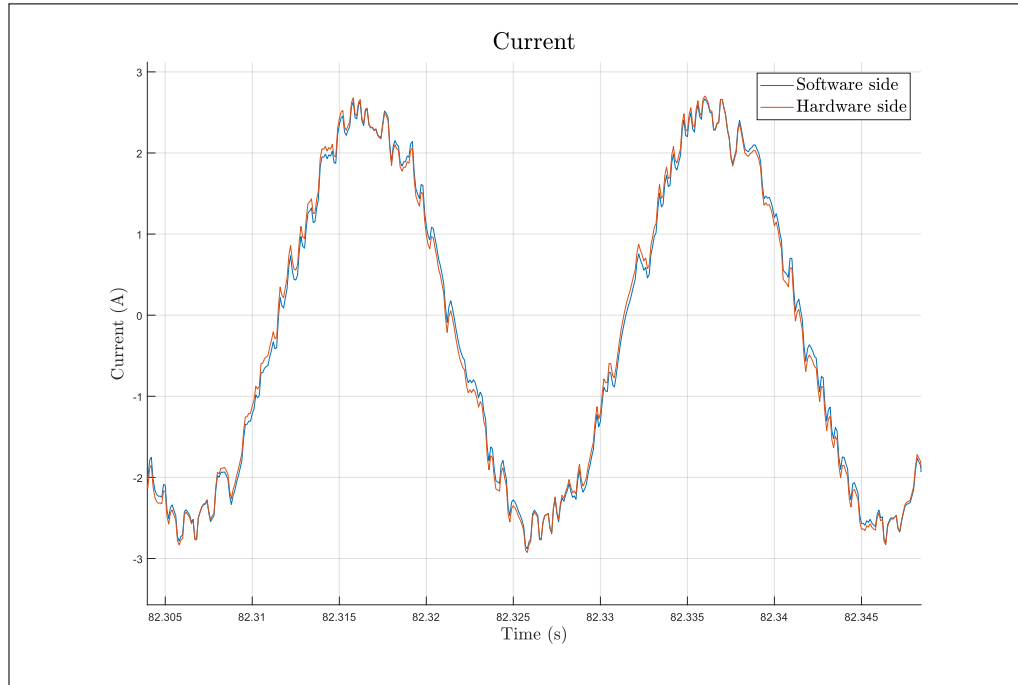


FIGURE 5.33. Current waveform before first Z_{DIM} update while feedback is on

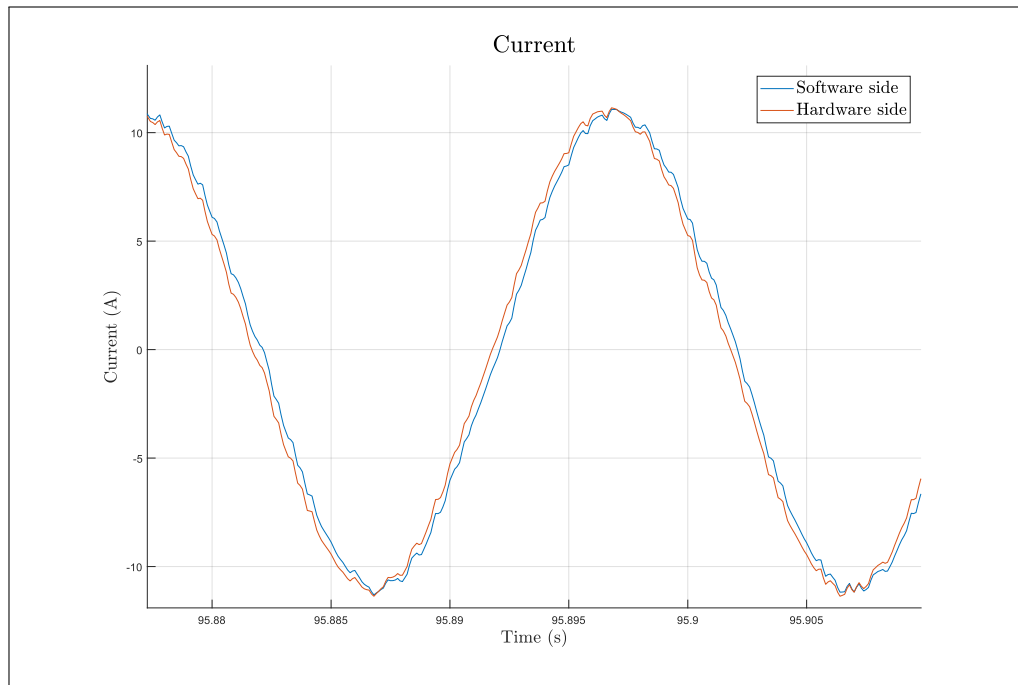


FIGURE 5.34. Current waveform after first Z_{DIM} update while feedback is on

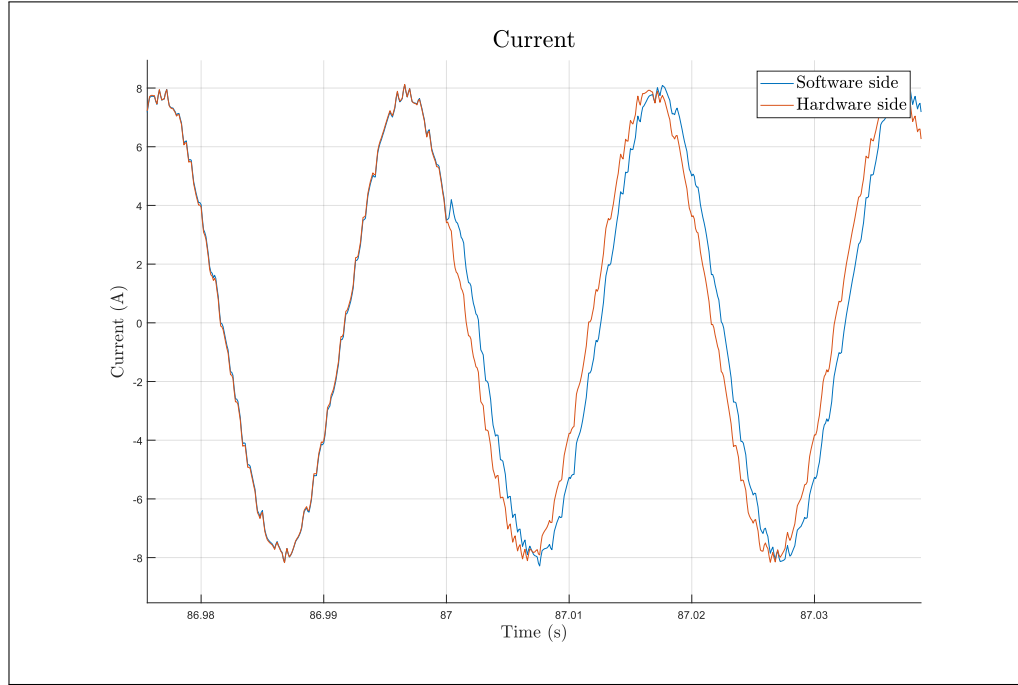


FIGURE 5.35. Current waveform during first Z_{DIM} update while feedback is on

A second test is performed using the same system as before, with one modification: after 30 seconds the module that implements Z_{DIM} will stop updating its parameters. To rule out any other difference between the two simulations, the rest of the identification process was kept running and its results discarded. This ensures that any improvements are only due to the change in Z_{DIM} and not due to not having to run the complex identification routine or having the PRBS disturbing the voltage of the connection point. The results of the test, shown in figures 5.38 and 5.39, confirm the changes were useful as hardware and software variables are matched. Further more, the phase error present in the current is gone, as seen in figure 5.42. This in turn proves our hypothesis that the cause of the inaccuracies was the feedback due to the inverter control messing the spectral analysis. Figures 5.40 and 5.41 show that hardware and software active and reactive power within 20W and 20 VAR respectively and remain accurate even during fast transients. The dynamics during transients is very important as it is indicative that even the behaviour of Z_{HUT} at higher

frequencies was captured during the identification process. The results seem to indicate that the identification process has been successful, and that the results can be trusted. This validates the approach stated in section 4.7. Furthermore, this achieved our initial goal, to have a PHIL platform capable of delivering accurate results without prior knowledge of the HUTs topology or control laws.

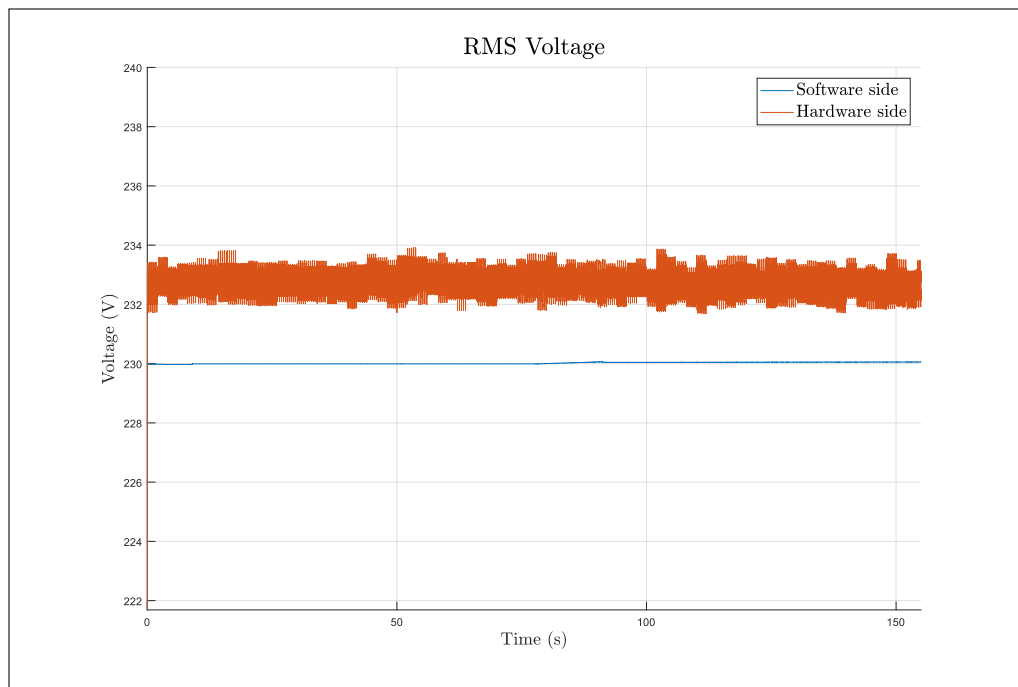


FIGURE 5.36. RMS voltage during second test

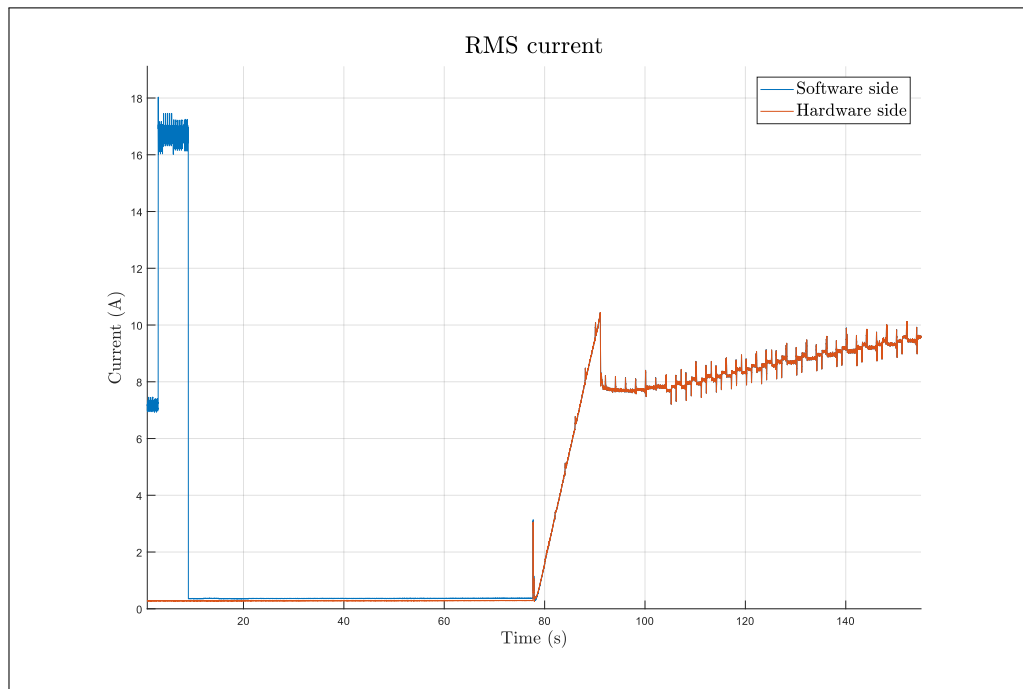


FIGURE 5.37. RMS current during second test

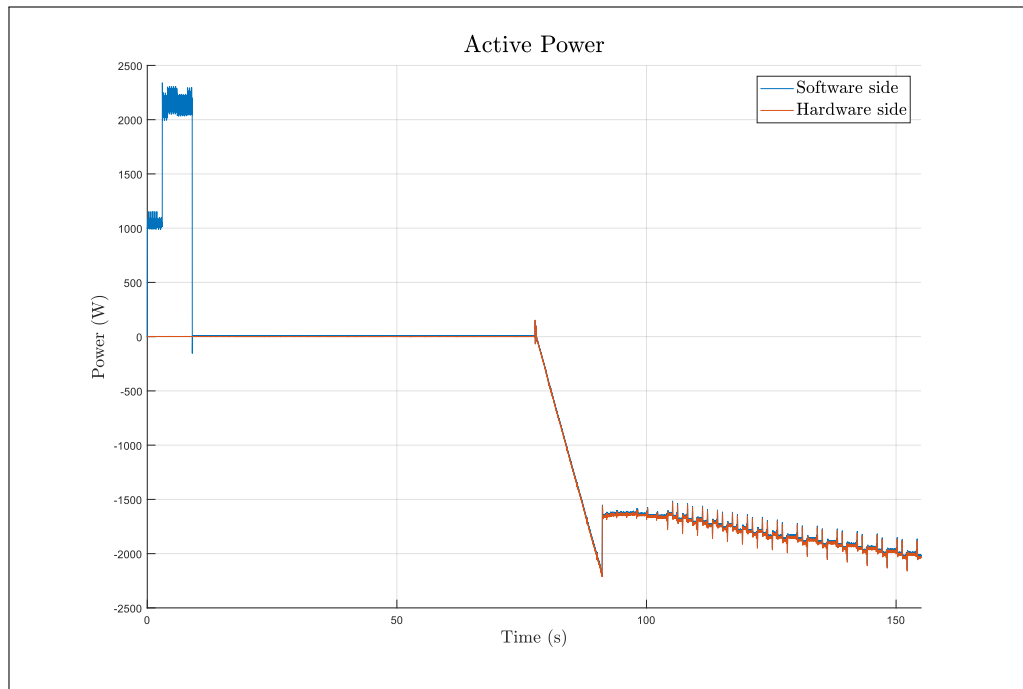


FIGURE 5.38. Active power during second test

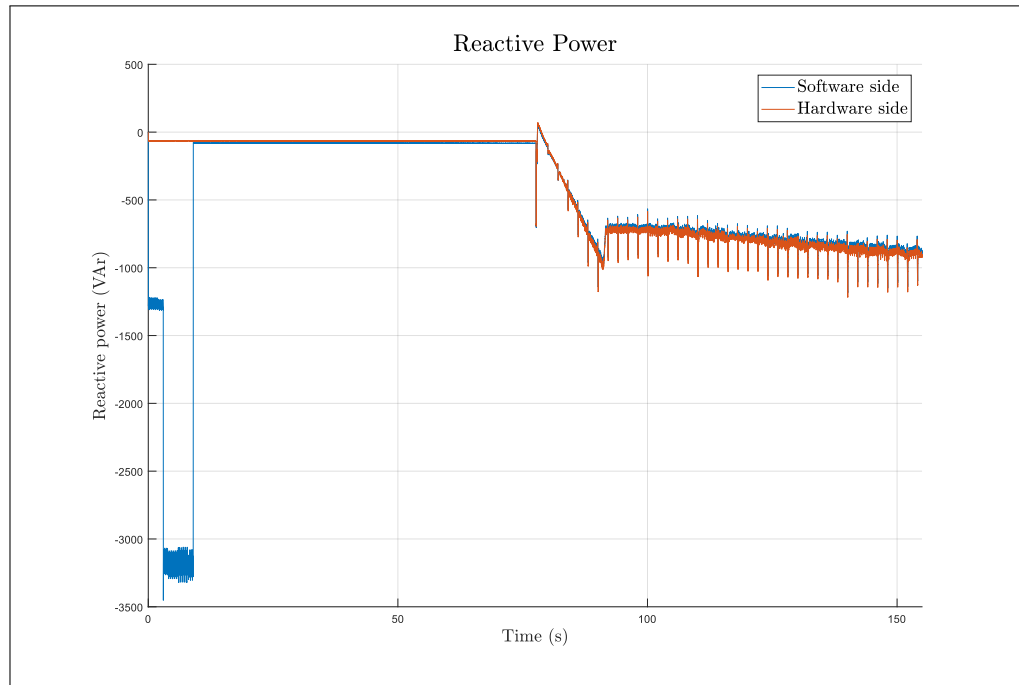


FIGURE 5.39. Reactive power during second test.

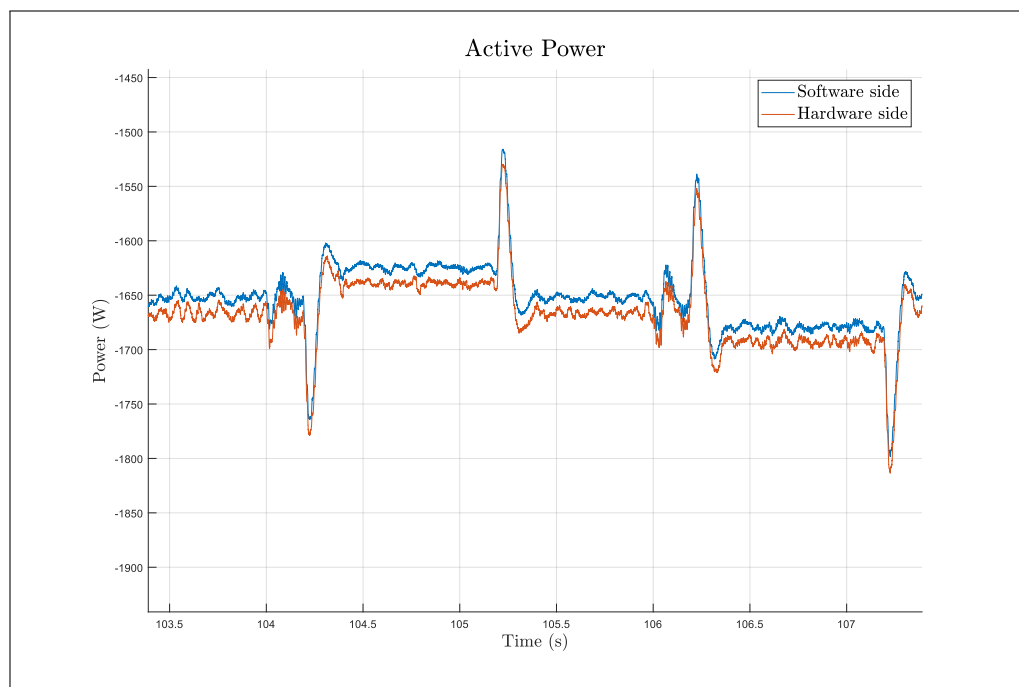


FIGURE 5.40. Active power during second test. Zoom

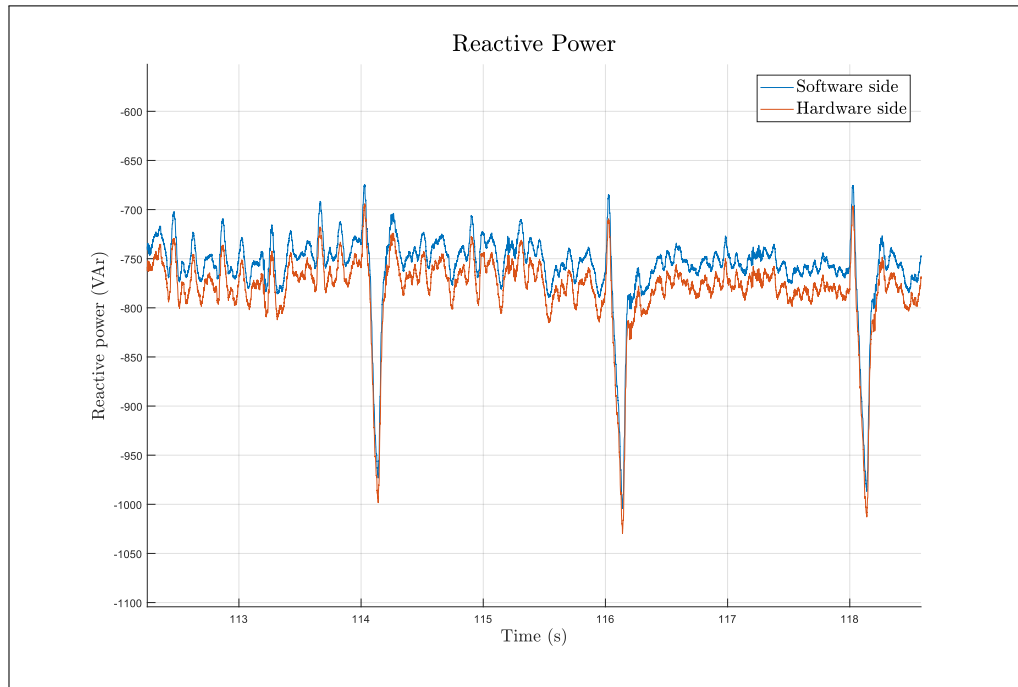


FIGURE 5.41. Reactive power second during test. Zoom

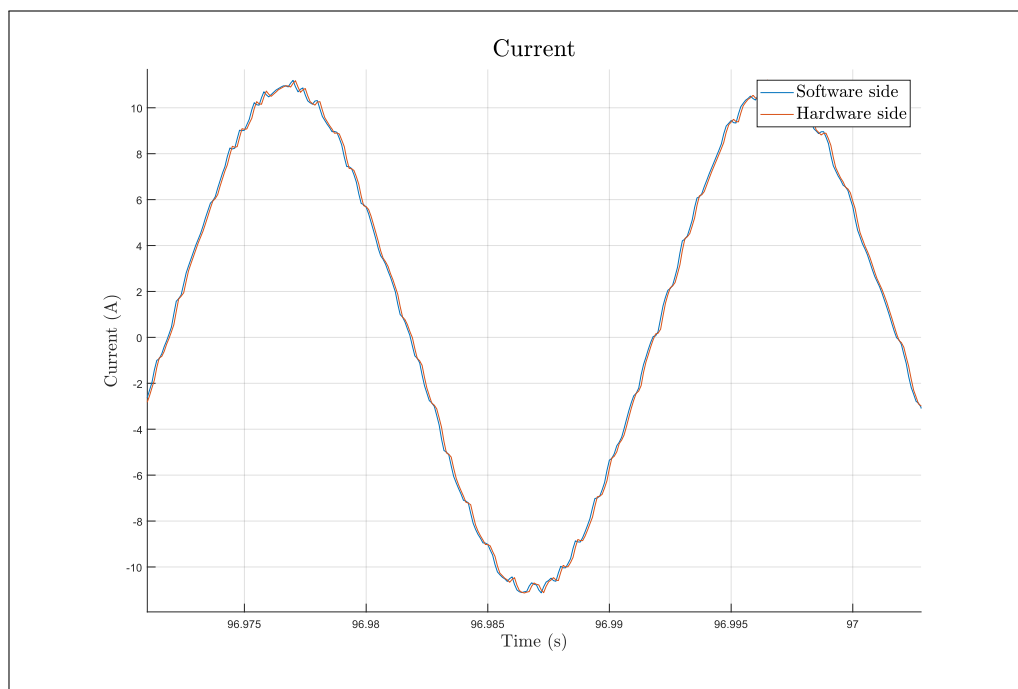


FIGURE 5.42. Current waveform with almost no phase error

5.7. Chapter conclusion

In this Chapter we presented the hardware implementation of our PHIL platform describing each of the components used, covering their capabilities and limitations. We also explained how the software model presented in Chapter 4 was modified to run in real time, as well as the considerations and rules one must take into account when programming something for real time simulation.

Finally we put our PHIL platform to the test, to validate our predictions from the previous Chapter. We see that results match our expectations, and that the platform performs well for the passive HUT case while failing to provide good results for the active HUT due to the interference's of the feedback in the identification process.

We also show that the solution proposed in section 4.7, of identifying the device's output impedance while the feedback is not present and then use that model for the rest of the simulation proved useful, allows for a stable and accurate simulation. However, this approach sacrifices the capability to re-identify the impedance during the simulation. This could impact accuracy in longer simulations were the electrical parameters of the output impedance vary due to heating or some other reason. It is also not a valid approach if the Z_{HUT} is expected to vary significantly during the simulation due to changes in its topology.

6. CONCLUSIONS

In chapter 2, we performed a thorough analysis of the different interface algorithms, showing that the DIM has the best performance in terms of stability and accuracy, if the damping impedance matches that of the HUT. In fact, when $Z_{HUT} = Z_{DIM}$, it became impossible for the system to become unstable. We also saw, how even relatively large mismatches between these two impedances, of up to 400 %, still managed to produce a stable simulation, in cases where the ITM proved unstable. The IA also managed to produce accurate results, provided the impedance matching was good. While even large impedance mismatches showed good accuracy at lower frequencies, to achieve accurate results throughout the frequency spectrum, good impedance matching was required. The ITM and ITM with low pass filter remain an option though, as their easy implementation can potentially save valuable time and resources. However they should only be considered if the conditions of the experiment allow for their use. Thus, the DIM remains the only interface algorithm capable of delivering successful PHIL simulations regardless of the impedances ratio between the HUT and the ROS. Because of this reliability, the DIM was the algorithm selected for the implementation of the PHIL platform.

Chapter 3 delves deep into the strategy selected to identify the impedance of the HUT. Encouraged by the good results showed in [Liegmann et al. \(2016\)](#) and [Riccobono et al. \(2017\)](#), we chose a strategy based on spectral analysis rather than simply rely on measuring magnitude and phase at a particular frequency. Testing of the identification routine showed good results, as it was capable of correctly identifying the spectrum of the tested systems across a wide range of frequencies and fit a parametric model of the appropriate order to them. Testing was done with different topologies and results were good either way.

Chapter 4 provides perhaps the biggest contribution of the thesis as we simulate the entire PHIL platform. We find that the methodology used, provided good results in cases with passive HUTs but failed when the HUT had an inner feedback loop that affects the variables used for the identification. In this cases, spectral analysis will instead provide

an estimate of the entire dynamics of the device, not just its output impedance, which is what is needed to ensure stability and accuracy, as shown in chapter 2. As we are unable to separate the dynamics of the output impedance from those of the rest of the system, our estimates for Z_{HUT} are flawed, leading to poor accuracy. It is worth noting that despite a poorly estimated Z_{HUT} , the simulations remained stable. This is to be expected as results in chapter 2 indicated that the DIM IA was very resilient to errors in the choice of Z_{DIM} in terms of stability. We then proceed to evaluate different approaches that can indeed be successful in identifying Z_{HUT} however we concluded that none of them were easy to apply. In the general case, only the indirect approach is applicable to identify the system, however this requires prior knowledge of the feedback loop of the device, in order to separate the two dynamics.

The results obtained through simulation are verified experimentally in chapter 5. For the case of an active HUT, we used a commercial residential inverter, whose control logic or output impedance were unknown to us. While our technique failed at first due to the reasons outlined in chapter 4, we were able to perform still able to perform a successful PHIL simulation with the device. Knowing it was the feedback loop of the inverter's control systems what caused the identification to fail, we took advantage of a period that the inverter uses to test for grid conditions before connection, to identify its output impedance without the influence of the control system. Once identified, re-identification was stopped and the simulation proceeded as normal. This allowed for a PHIL simulation with the same degree of accuracy as in the passive load case.

Given the results, we can say that only hypothesis H_1 can be confirmed, as it is indeed true that the DIM will provide the best performance possible of all the interface algorithms currently described in the literature. However, this is only guaranteed if the damping impedance matches the impedance of the HUT.

For hypothesis H_2 , results indicate that it is only partially true. The approach used will be successful only when the HUT does not have an internal feedback loop that can

interfere with the identification. In this cases the identification will provide accurate results and lead to a successful PHIL experiment. When internal feedback loop are present in the HUT, the results of the identification will provide an estimate for the entire system, not just the output impedance. This will compromise the simulations accuracy and potentially, even its stability.

With these findings in mind and given the challenges of continuously re-identifying the output impedance of an HUT with inner feedback loops, considerations should be made about whether this is really necessary during a PHIL simulation, or if identification at the beginning of the simulation is sufficient to ensure stability and accuracy. As such we leave as future work the identifications of the different scenarios where this would be possible and the ones where re-identification is crucial. We also leave as future work the identification of the methods that would allow, given knowledge of the feedback dynamics, provide us with an estimate of the HUT's impedance, when using the indirect method of identification.

Finally, we also leave as future work the testing of the platform with a broader range of devices, such as electric motors, rectifiers, statcoms or other types of active HUTs, to determine the cases where continuous re-identification is really necessary.

BIBLIOGRAPHY

- Ainsworth, N., Hariri, A., Prabakar, K., Pratt, A., y Baggu, M. (2016). Modeling and compensation design for a power hardware-in-the-loop simulation of an AC distribution system. En *Naps 2016 - 48th north american power symposium, proceedings*. doi: 10.1109/NAPS.2016.7747941
- Bañuelos, E., Gutiérrez, J. A., y Gustavsen, B. (2017a). Fitting methods. En *Rational fitting techniques for the modelling of electric power components and systems using matlab enviroment* (pp. 7–17). Intech. doi: 10.5772/intechopen.71358
- Bañuelos, E., Gutiérrez, J. A., y Gustavsen, B. (2017b). Matlab algorithms and applications. En *Rational fitting techniques for the modelling of electric power components and systems using matlab enviroment* (pp. 17–44). Intech. doi: 10.5772/intechopen.71358
- Benigni, A., Helmedag, A., Abdalrahman, A. M. E., Piłatowicz, G., y Monti, A. (2011). FlePS: A power interface for Power Hardware In the Loop. En *Proceedings of the 2011 14th european conference on power electronics and applications* (pp. 1–10).
- Berlekamp, E. R. (1968). *Algebraic coding theory*. New York: McGraw-Hill.
- Blackman, R. B., y Tukey, J. W. (1958). The Measurement of Power Spectra from the Point of View of Communications Engineering â Part I. *Bell System Technical Journal*. doi: 10.1002/j.1538-7305.1958.tb03874.x
- Brandl, R. (2017). *Operational range of several interface algorithms for different power hardware-in-the-loop setups*. doi: 10.3390/en10121946

Brillinger, D. R. (1981). The estimation of power spectra. En *Time series: Data analysis and theory* (Second ed., pp. 116–185). San Francisco: Holden Day.

Delta elektronika B.V. (2020). *SM15K - series* (Inf. Téc.). Zierikzee: Delta elektronika. Descargado de <https://www.delta-elektronika.nl/upload/PRODUCT{ }MANUAL{ }SM15K{ }P0110{ }V201908.pdf>

Fowler, K. (2015). V-Model. En *Developing and managing embedded systems and products* (cap. 1).

Grogan, P. T., De Weck, O. L., Ross, A. M., y Rhodes, D. H. (2015). Interactive models as a system design tool: Applications to system project management. En *Procedia computer science*. doi: 10.1016/j.procs.2015.03.015

Hatakeyama, T., Riccobono, A., y Monti, A. (2016). Stability and accuracy analysis of power hardware in the loop system with different interface algorithms. En *2016 IEEE 17th workshop on control and modeling for power electronics, compel 2016*. doi: 10.1109/COMPEL.2016.7556671

Huawei. (2016). *SUN2000L-2/3/3.68/4/4.6/5KTL* (Inf. Téc.). Shenzhen, China. Descargado de <https://www.solaryours.com/wp-content/uploads/2018/11/SUN2000L.pdf>

IEC. (2010). *Communication networks and systems for power utility automation - Part 7-4: Basic communication structure - Compatible logical node classes and data object classes*.

IEEE Standard Association. (2011). *IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads*. doi: 10.1109/IEEESTD.2011.6018239

IEEE Standard Association. (2018). *IEEE Std. 1547-2018. Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces*. doi: 10.1109/IEEESTD.2018.8332112

IEEE Standard Association. (2020, may). IEEE Standard Conformance Test Procedures for Equipment Interconnecting Distributed Energy Resources with Electric Power Systems and Associated Interfaces. *IEEE Std 1547.1-2020*, 1–282. doi: 10.1109/IEEESTD.2020.9097534

Iwado, N., Ohori, A., Hattori, N., y Funaki, T. (2015, oct). Stabilization techniques of power hardware-in-the-loop simulation with time delay compensation. En *2015 ieee international telecommunications energy conference (intelec)* (pp. 1–5). doi: 10.1109/INTLEC.2015.7572293

Jiang, S., Li, G., Xin, Y., Wang, L., y Wang, W. (2019). Interface algorithm development for PHIL simulations of MMC-HVDC devices via real-time impedance matching. *The Journal of Engineering*. doi: 10.1049/joe.2018.8691

Karapanos, V., De Haan, S., y Zwetsloot, K. (2011). Real time simulation of a power system with VSG hardware in the loop. En *Iecon proceedings (industrial electronics conference)*. doi: 10.1109/IECON.2011.6119919

Langston, J., Schoder, K., Steurer, M., Faruque, O., Hauer, J., Bogdan, F., ... Katiraei, F. (2012). Power hardware-in-the-loop testing of a 500 kW photovoltaic array inverter. En *Iecon proceedings (industrial electronics conference)*. doi: 10.1109/IECON.2012.6389595

Lehfuss, F., Lauss, G., Kotsampopoulos, P., Hatziargyriou, N., Crolla, P., y Roscoe, A. (2012). Comparison of multiple power amplification types for power Hardware-in-the-Loop applications. En *2012 ieee workshop on complexity in engineering, compeng 2012 - proceedings*. doi: 10.1109/CompEng.2012.6242959

- Lehfuß, F., Lauss, G., y Strasser, T. (2012). Implementation of a multi-rating interface for Power-Hardware-in-the-Loop simulations. En *Iecon proceedings (industrial electronics conference)*. doi: 10.1109/IECON.2012.6389004
- Lemaire, M., Sicard, P., y Belanger, J. (2015). Prototyping and Testing Power Electronics Systems Using Controller Hardware-In-the-Loop (HIL) and Power Hardware-In-the-Loop (PHIL) Simulations. En *2015 ieee vehicle power and propulsion conference, vppc 2015 - proceedings*. doi: 10.1109/VPPC.2015.7353000
- Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*. doi: 10.1090/qam/10666
- Liegmann, E., Riccobono, A., y Monti, A. (2016). Wideband identification of impedance to improve accuracy and stability of power-hardware-in-the-loop simulations. En *2016 ieee international workshop on applied measurements for power systems, amps 2016 - proceedings*. doi: 10.1109/AMPS.2016.7602873
- Liu, Y., Steurer, M., y Ribeiro, P. (2005). A novel approach to power quality assessment: Real time hardware-in-the-loop test bed. *IEEE Transactions on Power Delivery*. doi: 10.1109/TPWRD.2005.844251
- Ljung, L. (1999). Nonparametric time - and - frequency-domain methods. En *System identification: Theory for the user* (pp. 168–197). Upper saddle river, New Jersey: Prentice Hall.
- Marks, N. D., Kong, W. Y., y Birt, D. S. (2018a). Interface Compensation for Power Hardware-in-the-Loop. En *Ieee international symposium on industrial electronics*. doi: 10.1109/ISIE.2018.8433620

Marks, N. D., Kong, W. Y., y Birt, D. S. (2018b). Stability of a switched mode power amplifier interface for power hardware-in-the-loop. *IEEE Transactions on Industrial Electronics*. doi: 10.1109/TIE.2018.2814011

Marquardt, D. W. (1963). An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*. doi: 10.1137/0111030

Mathworks. (2020). *Functions and Objects Supported for C/C++ Code Generation*. Descargado 2020-04-29, de <https://la.mathworks.com/help/coder/ug/functions-and-objects-supported-for-cc-code-generation.html>

Nassif, A. B. (2018). An Analytical Assessment of Feeder Overcurrent Protection with Large Penetration of Distributed Energy Resources. *IEEE Transactions on Industry Applications*. doi: 10.1109/TIA.2018.2810260

Newton, A. R., y Sangiovanni-Vincentelli, A. L. (1984). Relaxation-Based Electrical Simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. doi: 10.1109/TCAD.1984.1270089

OPAL-RT. (2020a). *OP5600V2 System description*. Descargado 2020-04-29, de <https://wiki.opal-rt.com/display/HDGD/OP5600V2>

OPAL-RT. (2020b). *OPAL-RT resource center*. Descargado de <https://www.opal-rt.com/resource-center/>

Paran, S. (2013). *Utilization of Impedance Matching to Improve Damping Impedance Method- Based Phil Interface*. Florida State University Libraries.

Paran, S., y Edrington, C. S. (2013). Improved power hardware in the loop interface methods via impedance matching. En *2013 IEEE Electric Ship Technologies Symposium, ESTS 2013*. doi: 10.1109/ESTS.2013.6523758

Paran, S., Fleming, F., Li, D., y Edrington, C. S. (2014). Utilization of adaptive PHIL interfaces for harmonic load cases. En *Iecon proceedings (industrial electronics conference)*. doi: 10.1109/IECON.2014.7049066

Parise, G. (2013). A new summary on the IEC protection against electric shock. *IEEE Transactions on Industry Applications*. doi: 10.1109/TIA.2013.2244547

Peng, X., y Wild, J. (2017). Innovative Microgrid Solution for Renewable Energy Integration within the REIDS Initiative. En *Energy procedia*. doi: 10.1016/j.egypro.2017.12.733

Petersen, K., Wohlin, C., y Baca, D. (2009). The waterfall model in large-scale development. En *Lecture notes in business information processing*. doi: 10.1007/978-3-642-02152-7_29

Puissance Plus. (2020). *4Q Power Amplifiers AC - DC - Three - Phase - 3x7000VA* (Inf. Téc.). Mountauban: Puissance Plus. Descargado de <https://www.puissanceplus.com/assets/produitspdf/Amplifier4QACDCThreephases3x7kVAlimitedabsorptionV7.pdf>

Radatz, P., Kagan, N., Rocha, C., Smith, J., y Dugan, R. C. (2016). Assessing maximum DG penetration levels in a real distribution feeder by using OpenDSS. En *Proceedings of international conference on harmonics and quality of power, ichqp*. doi: 10.1109/ICHQP.2016.7783416

Ren, W. (2007). *Accuracy Evalaution of Power Hardware-in-the- Loop (PHIL) Simulation*. Florida State University Libraries.

Ren, W., Steurer, M., y Woodruff, S. (2007). Applying controller and power hardware-in-the-loop simulation in designing and prototyping apparatuses for future all electric ship. En *Ieee electric ship technologies symposium, ests 2007*. doi: 10.1109/ESTS.2007.372124

Riccobono, A., Liegmann, E., Pau, M., Ponci, F., y Monti, A. (2017). Online Parametric Identification of Power Impedances to Improve Stability and Accuracy of Power Hardware-in-the-Loop Simulations. *IEEE Transactions on Instrumentation and Measurement*. doi: 10.1109/TIM.2017.2706458

Seo, H. R., Park, M., Yu, I. K., y Song, B. M. (2011). Performance analysis and evaluation of a multifunctional grid-connected PV system using power hardware-in-the-loop simulation. En *Conference proceedings - ieee applied power electronics conference and exposition - apec*. doi: 10.1109/APEC.2011.5744862

Siegers, J., y Santi, E. (2014). Improved power hardware-in-the-loop interface algorithm using wideband system identification. En *Conference proceedings - ieee applied power electronics conference and exposition - apec*. doi: 10.1109/APEC.2014.6803459

Steurer, M., Bogdan, F., Ren, W., Sloderbeck, M., y Woodruff, S. (2007). Controller and power hardware-in-loop methods for accelerating renewable energy integration. En *2007 ieee power engineering society general meeting, pes*. doi: 10.1109/PES.2007.386022

Stoica, P., y Moses, R. (2005). *Spectral analysis of Signals*. Upper saddle river, New Jersey: Prentice Hall.

Torsten, S., y Stoica, P. (1989a). Identification of systems operating in a closed loop. En *Systems identification* (pp. 381–416). Cambirdge, UK: Prentice Hall.

- Torsten, S., y Stoica, P. (1989b). Input signals. En *Systems identification* (pp. 96–137). Cambirdge, UK: Prentice Hall.
- Torsten, S., y Stoica, P. (1989c). Nonparametric methods. En *Systems identification* (First ed., pp. 32–58). Cambirdge, UK: Prentice Hall.
- Wang, W., y De Leon, F. (2020). Quantitative Evaluation of der Smart Inverters for the Mitigation of FIDVR in Distribution Systems. *IEEE Transactions on Power Delivery*. doi: 10.1109/TPWRD.2019.2929547
- Wei, L. (2017). *Tutorial: RT-LAB for Real-Time Simulation Applications in Power Electronics* (Inf. Téc.). Montreal, Canada: OPAL-RT technologies.
- Weisstein, E. W. (2020). *Primitive polynomials*. Descargado 2020-04-26, de <https://mathworld.wolfram.com/PrimitivePolynomial.html>
- Welch, P. D. (1967). The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms. *IEEE Transactions on Audio and Electroacoustics*. doi: 10.1109/TAU.1967.1161901
- Wu, X., Lentijo, S., y Monti, A. (2004). A novel interface for Power-Hardware-In-the-Loop simulation. En *Proceedings of the ieee workshop on computers in power electronics, compel*. doi: 10.1109/CIPE.2004.1428147
- Wu, X., y Monti, A. (2005). Methods for partitioning the system and performance evaluation in power-hardware-in-the-loop simulations - Part I. En *Iecon proceedings (industrial electronics conference)*. doi: 10.1109/IECON.2005.1568912
- Zhang, Z. (2016). *Power Hardware in-the-loop test system*. Graz: Bibliothek der TU Graz.

Zyuzev, A. M., Mudrov, M. V., y Nesterov, K. E. (2016, oct). PHIL-system for electric drives application. En *2016 ix international conference on power drives systems (icpds)* (pp. 1–4). doi: 10.1109/ICPDS.2016.7756687

ANEXO

ANEXO A. DEVICE TECHNICAL INFORMATION

SUN2000L-2/3/3.68/4/4.6/5KTL

Technical Specification

Technical Specification	SUN2000L-2KTL	SUN2000L-3KTL	SUN2000L-3.68KTL	SUN2000L-4KTL	SUN2000L-4.6KTL	SUN2000L-5KTL
Efficiency						
Max. efficiency	98.4 %	98.5 %	98.5 %	98.6 %	98.6 %	98.6 %
European weighted efficiency	97.0 %	97.6 %	97.8 %	97.9 %	98.0 %	98.0 %
Input						
Recommended max. PV power	3,000 Wp	4,500 Wp	5,520 Wp	6,000 Wp	6,900 Wp	7,500 Wp
Max. input voltage	600 V / 495 V ¹					
Operating voltage range ¹	90 V~ 600 V / 90 V~ 495 V ¹					
Start-up voltage	120 V					
Full power MPPT voltage range	120 V ~ 480 V	160 V ~ 480 V	190 V ~ 480 V	210 V ~ 480 V	260 V ~ 480 V	260 V ~ 480 V
Rated input voltage	380 V					
Max. input current per MPPT	11 A					
Max. short-circuit current	15 A					
Number of MPP trackers	2					
Max. number of inputs	2					
Output						
Grid connection	Single phase					
Rated output power	2,000 W	3,000 W	3,680 W	4,000 W	4,600 W	5,000 W ²
Max. apparent power	2,200 VA	3,300 VA	3,680 VA	4,400 VA	5,000 VA ³	5,500 VA ⁴
Rated output voltage	220 V / 230 V / 240 V					
Rated AC grid frequency	50 Hz / 60 Hz					
Max. output current	10 A	15 A	16 A	20 A	23 A ⁵	25 A ⁵
Adjustable power factor	0.8 leading ... 0.8 lagging					
Max. total harmonic distortion	≤ 3 %					
Protection						
Anti-islanding protection	Yes					
DC reverse polarity protection	Yes					
Insulation monitoring	Yes					
DC surge protection ⁶	Yes					
AC surge protection ⁶	Yes					
Residual current monitoring	Yes					
AC overcurrent protection	Yes					
AC short-circuit protection	Yes					
AC overvoltage protection	Yes					
Over-heat protection	Yes					
General Data						
Operating temperature range	-30 ~ +60 °C (Derating above 45°C @ Rated output power)					
Relative operating humidity	0 %RH ~ 100 %RH					
Operating altitude	0 ~ 4,000 m (Derating above 2,000 m)					
Cooling	Natural convection					
Display	LED indicators					
Communication	RS485, WLAN					
Weight (incl. mounting bracket)	10.6 kg (23.4 lb)					
Dimension (incl. mounting bracket)	375 x 375 x 161.5 mm (14.8 x 14.8 x 6.4 inch)					
Degree of protection	IP65					
Nighttime Power Consumption	< 2 W					
Battery Compatibility						
Battery	LG Chem RESU 7H_R / 10H_R					
Voltage range	350 ~ 450 Vdc					
Max. current	10 A					
Communication	RS485					
Optimizer Compatibility						
DC MBUS compatible optimizer	SUN2000P-375W via Smart PV Safety Box SmartPSB2000L					
Standard Compliance (more available upon request)						
Safety	EN/IEC 62109-1, EN/IEC 62109-2					
Grid connection standards	GB3/2, G59/3, EN 50438, CEI 0-21, VDE-AR-N-4105, AS 4777, C10/11, ABNT, UTE C15-712, RD 1699, NRS 097-2-1, DEWA 2016					

1. Photo-voltaic (PV) modules. The maximum input voltage and maximum output current must not exceed the 600 Vdc and 15 A values respectively. 2. With 100 °C ambient temperature. 3. With 100 °C ambient temperature. 4. With 100 °C ambient temperature. 5. With 100 °C ambient temperature. 6. With 100 °C ambient temperature.

FIGURE A.1. Huawei inverter technical data

OUTPUTS: POWER								
Power								
Rated power per phase	7 000VA							
Rated power total	21 000VA							
AC ranges	130V / 200V / 260V / 400V / 520V / 800V							
DC ranges	180V / 280V / 360V / 560 V / 720V / 1120V							
Output type	Direct (without transformer)							
Voltage and current in AC (RMS values)	Low voltage ranges				High voltage ranges (1)			
	135V	200V	270V	400V	270V	400V	540V	800V
AC voltage (VRMS)	0-135	0-200	0-270	0-400	0-270	0-400	0-540	0-800
Permanent AC current "In"	0-54	0-36	0-28	0-18	0-54	0-36	0-28	0-18
Maximum peak current (1)	162	108	84	54	162	108	84	51
Voltage and current in DC	Low voltage ranges				High voltage ranges (1)			
	190V	280V	380V	560V	380V	560V	760V	1120V
DC voltage	±190	±280	±380	±560	±360	±560	±720	±1120
Permanent DC current "In"	±54	±36	±28	±18	±54	±36	±28	±18
Maximum peak current (1)	162	108	84	54	162	108	84	54

FIGURE A.2. Puissance Plus power specification

OUTPUTS: RESOLUTION / ACCURACY	
Voltage accuracy	
Typical	0,1% of range + 0,1% of programmed value
Resolution	12 bits
Current accuracy	
Typical	0,1% of range + 0,1% of programmed value
Resolution	12 bits
Voltage distortion at full output power	
Typical	< 0,3% (max < 0.7%)
Voltage regulation for a mains variation of +6% / -10%	
Max	< 0,1% of rated voltage
Voltage regulation for a current variation from 0 to 100%	
Max	< 0,1% of rated voltage
Noise	
Max RMS	0,02% of rated voltage
Max peak to peak	0,3% of rated voltage
Variation according temperature	
Typical	50 ppm/°C (max 100 ppm/°C)
Stability after 15 minutes of operation	
Max	< 0,05% of rated voltage
Insulation of the outputs versus case ground	
Measurement at 500 VDC	> 100 MΩ

FIGURE A.3. Puissance Plus accuracy specification

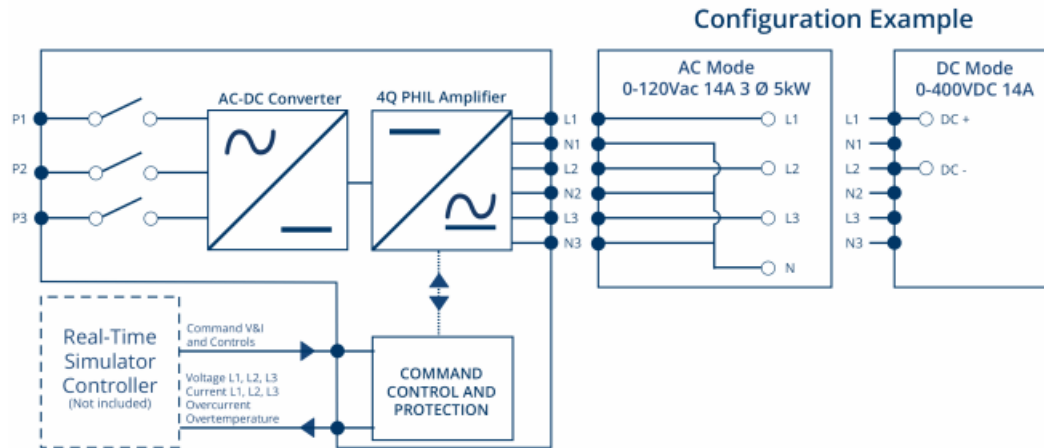
TIME FEATURES	
Output signal (internal generator)	
Frequency	DC or 40 Hz to 2 kHz Resolution 0.1 Hz
Dephasing	$\pm 360^\circ$ Resolution 0.1°
Amplitude	0 to 7,07 VRMS $\pm 10V$ peak
Output bandwidth	
Full scale	DC – 15 kHz
Small signals at -3 dB	70 kHz
Output variation with a square signal pilot (3)	
Rise time 10% / 90%	< 7 μ s (voltage regulation) < 100 μ s (current regulation)
Fall time 10% / 90%	< 7 μ s (voltage regulation) < 100 μ s (current regulation)
Transfer time	< 7 μ s (voltage regulation) < 100 μ s (current regulation)
Transition from Q1 to Q4	< 10 μ s

FIGURE A.4. Puissance Plus time specification

MECANICAL AND ENVIRONMENTAL	
Material and surface treatment	
Frame	Aluminum painted RAL7021
Sides and rear panels	Aluminum painted RAL7021
Dimensions and weight	
Width	800 mm
Depth	800 mm
Height	1950 mm (38U)
Total weight (6)	550 kg
Handling	
Width	Four wheels d 125 mm with brakes
Temperature and humidity	
Stockage temperature	-10°C à +85°C
Operation temperature	+0°C à +50°C
Humidity	10% - 90% non-condensing
Noise (fans at full speed)	
Measured at 1 m	< 70 dBA
Marking	
Marking	CE
Protection	IP20

FIGURE A.5. Puissance Plus physical specifications

OP1400-10 | TYPICAL DIAGRAM 5KW / 120V



POWER MODULE SPECIFICATIONS

CHARACTERISTIC	OP1400-10	OP1400-20	OP1400-30	
	Configuration for 120Vrms			Configuration for 240Vrms
AC Mode Voltage Range (L-N)	0-120/208VRMS			0-240V/380Vrms
Total nominal Power	5KW at 120Vrms	10KW at 120Vrms	15KW at 120Vrms	10KW at 240Vrms
Maximum number of phase	3 phases	6 phases	9 phases	3 phases
DC Mode Voltage Range (DC+ DC-)	+/- 400VDC 1 DC ouput	+/- 400VDC 2 DC ouputs	+/- 400VDC 3 DC ouputs	
Current Range per phase	0-14 Arms			
Current Peak	20Apk			
Bandwith (Hz)	DC to 10 KHZ (-3db)			
Absorption capacity	100%, power regenerated, no dissipation			
Cooling	Air forced			
Efficiency	90%			
THD (3dB)	0.5% @ 0 - 1kHz / 1% @ 1kHz - 2kHz / 2% @ 2kHz - 10kHz			
Slew Rate	5V/us, independant of the load			
Time delay Input to ouput	5.5us to 8.3us			

FIGURE A.6. OP1400 amplifier technical data