# Acceleration of Bayesian Model Based Data Analysis

Dissertation to obtain the degree of
Doctor of Engineering (Dr.-Ing.) of the
Faculty of Computer Science and Electrical Engineering
University of Rostock

Reviewed by:
Prof. Dr.-Ing Dirk Timmermann, University of Rostock
Prof. Dr. Robert C. Wolf, Max Planck Institute for Plasma Physics
Dr. Jakob Svensson, Max Planck Institute for Plasma Physics

Submitted by
Humberto Trimiño Mora, born on April 4, 1987 in San José, Costa Rica
from Greifswald

Rostock, November 29, 2018
Year of defense: 2019

https://doi.org/10.18453/rosdok_id00002491

# Humberto Trimiño Mora

## Contact

Max Planck Institute for Physics

Wendelsteinstrasse 1

17489, Greifswald

Phone: (+49)03834 88 1938

Email: humtm@ipp.mpg.de

## Personal

Born on April 4, 1987.

San José, Costa Rica.

## Research projects and education

**Miniaturized device for failure analysis of integrated circuits, 2008.**
Verilog Code simulation of processing unit.

**Design of SCR-1 ECRH heating system, 2011**
Technological Institute of Costa Rica (ITCR)

**Licenciatura (Diplom/Lic.-Eng.) of Electronics Engineering, 2012.**
Technological Institute of Costa Rica (ITCR)
Thesis: Design and Implementation of the Signal Processing and Data Acquisition System for the W7-X Dispersion Interferometer

# Abstract

Inverse problems involving parameter estimation often face a choice between the use of a real-time scheme with strong approximations or rigorous post-processing with explicit uncertainty handling. Plasma physics experiments set a particularly high demand of both. Real-time control systems often need to rely on distinct and uncertain input data where knowing the estimated parameter uncertainty is crucial. Additionally, more rigorous scientific inference is required for rapid decision-making after experiments. Finally, it is also common that two or more diagnostics measure the same plasma parameter or correlated parameters, but their data is analyzed separately. A solution that meets all of these requirements is missing.

With Bayesian analysis it is possible to carry out a quantitative assessment of a parameter and its uncertainty. The joint analysis of data from several plasma diagnostics in a strict mathematical way is simplified and it improves the inference on correlated parameters. This makes standard Bayesian analysis an excellent tool for the case at hand, with the disadvantage of extensive processing times, which are far too long for a real-time analysis.

Current variations of Bayesian analysis like Kalman filters or sub-optimal Bayesian online algorithms target specific cases of this problem. But for most plasma physics time-independent non-linear inverse problems, the best option is to use a general approach of Bayesian analysis as done with several Wendelstein 7-X diagnostics. This approach avoids approximations when dealing with post-processing.

The work in this thesis shows how current Field Programmable Gate Array's (FPGA) resource and parallelization capabilities allow for an acceleration of this type of mathematically intense Bayesian analysis. It presents a significant acceleration in the estimation of plasma parameters and joint analysis of two important plasma diagnostics. Using deep pipelining and high parallelization, it maintains the required precision while reducing the processing time of the inversion of a mathematically demanding model. This work therefore presents a solution that satisfies the scientific experiment requirements while reducing the need for a speed vs. rigorosity trade-off.

# Kurzfassung

Die Bestimmung von Parametern bei inversen Problemen beinhaltet eine Abwägung zwischen der Anwendung von Echtzeitverfahren unter Verwendung vereinfachender Annahmen einerseits und rigoroser Datenanalyse mit expliziter Fehlerbetrachtung andererseits. Experimente in der Plasmaphysik stellen besonders hohe Anforderungen an beide. Echtzeitkontrollsysteme müssen oft auf unähnliche und ungenaue Eingangsdaten zurückgreifen, bei denen die Bestimmung der Unsicherheit besonders wichtig ist. Eine detailliertere, aber dennoch schnellere Auswertung nach Experimenten ist entscheidend, um über das weitere Vorgehen entscheiden zu können. Es ist auch typisch, dass mehrere Diagnostiken dieselben oder stark korrelierte Plasmaparameter messen, aber ihre Daten unabhängig voneinander ausgewertet werden. Eine Lösung, die alle diese Anforderungen erfüllt, fehlt.

Mit der Bayesschen Analyse ist es möglich eine quantitative Abschätzung eines Parameters und seiner Ungenauigkeit zu erhalten. Die gemeinsame Analyse von Daten mehrerer Diagnostiken auf mathematisch fundierte Weise wird durch sie vereinfacht und die Inferenz korrelierter Parameter verbessert. Die Bayessche Analyse ist deswegen ein ausgezeichnetes Werkzeug für diese Problemstellung, mit dem Nachteil einer langen Bearbeitungszeit, die weit entfernt ist von einer Echtzeitanalyse.

Spezialfälle dieses Problems werden aktuell durch Variationen Bayesscher Analyse wie Kalmanfilter oder suboptimale zeitgleiche Bayessche Algorithmen behandelt. Für die zeitunabhängigen, nichtlinearen, inversen Probleme der Plasmaphysik ist ein allgemeiner Ansatz die beste Option, wie er bereits bei vielen Diagnostiken in Wendelstein 7-X verwendet wird. Dieser Ansatz vermeidet Annäherungen in der Nachbearbeitung.

Diese Arbeit zeigt wie moderne Field Programmable Gate Arrays (FPGA) durch ihre Ressourcen und Parallelisierungsfähigkeiten die Beschleunigung dieser Art mathematisch aufwändiger Bayesscher Analyse ermöglichen. Sie beschreibt eine signifikante Beschleunigung in der Abschätzung von Plasmaparametern und die gemeinsame Analyse zweier wichtiger Plasmadiagnostiken. Durch die Verwendung von tiefem Pipelining und hoher Parallelisierung bei der Inversion eines mathemathisch komplexen Models wird die geforderte Präzision gewährleistet

und die erforderliche Rechenzeit reduziert. Somit stellt diese Arbeit eine Lösung dar, die den wissenschaftlichen Experimentanforderungen entspricht und die Notwendigkeit der erwähnten Geschwindigkeit vs. Rigorosität Abwägung reduziert.

# Contents

# List of Figures

# Declaration

I declare that this thesis was composed independently and that it does not incorporate, without acknowledgment, any material previously submitted for a degree in any university. To the best of my knowledge this thesis does not contain any materials previously published or written by any other person except where due reference is made.

Humberto Trimiño Mora

29th of November 2018

## Acknowledgments

First and foremost, I would like to thank my two advisors, Prof. Timmermann and Prof. Wolf. who patiently challenged me, teaching me how to produce, critically evaluate and share engineering and scientific knowledge. Their view on proper and thorough investigation sets a reference point that will continuously shape the way I work.

Within this process of further academic growth, it is important for me to credit Dr. Werner for giving me an opportunity to prove myself and making sure his personal and professional advice was always available. Through it, I learned how to identify crucial elements in a problem and avoid premature optimization. Together with Dr. Svensson, their views awoke my interest in Bayesian analysis in a scientific as well as a philosophical way.

I am also immensely grateful with the massive group of scientists, coworkers and friends at the IPP Greifswald, whose help and opinion were never unavailable. Special cases like the constant advising of Oliver Ford and Sebastian Tallents, profoundly contributed to my development of critical thinking.

Finally, it is paramount to denote that my interest for research and science is driven by the unabating motivation of my parents and core family. Their encouragement to ask, discover and learn about anything my curiosity would point to, made the attempt to follow this path a reality. Along the same lines but later in life came the constant support from my life partner

and new family, who through thick and thin stood by my side in front of every incoming life event.

## Thesis Overview

This thesis studies the acceleration feasibility through reconfigurable hardware of Bayesian data analysis of inverse problems, as they appear in complex physics experiments. The main aim is to solve complex inverse problems that are normally dealt with in a post processing scheme, on a time scale which approaches real-time.

The frame of this work is in the field of plasma physics. Here an estimation of important plasma parameters is required in a faster time frame for the control of the Wendelstein 7-X, without affecting inference through the introduction of approximations. For the proper analysis of the problem and the possibilities of acceleration the work was split into two sections. An initial stage with a simpler problem of low dimensionality and a second stage with a realistic, higher dimensionality problem.

Chapter 1 introduces the basic concepts of plasma physics, magnetic confinement devices and field programmable gate arrays which are the background of this thesis.

Chapter 2 defines and describes algorithms, tools and data analysis strategies used to carry out this work as well as describing the central method used in this thesis, Bayesian analysis. It also contains a detailed description of a tool developed in Python for optimizing the implementation of extensive arithmetic functions in hardware.

In Chapter 3, the feasibility of acceleration is studied in a realistic yet simplified case. The non-linear inverse problem of the Dispersion interferometer is analyzed with a single free parameter, limiting its dimensionality. This chapter describes the diagnostic design, the description of the diagnostic's model, the software analysis and the hardware implementation of the analysis.

Based on the results and the acceleration found in chapter 3, chapter 4 is the logical extension towards a problem with higher dimensionality and complexity. This chapter covers a new diagnostic with a more demanding model and a higher number of free parameters. Taking advantage of the Bayesian analysis, the joint analysis of two diagnostic models that share a parameter (as is the case with Thomson scattering and the Dispersion interferometer) is demonstrated. Similar to the previous chapter, the complete process from diagnostic design to hardware implementation of the model is presented.

Finally, chapter 5 discusses the results obtained in the previous chapters in terms of feasibility, acceleration, and precision. Since this project was separated in two representative cases of a single problem, a general analysis can't be carried out on each section. For this reason, the conclusions section also contains an in depth analysis of the results of each main section, followed by a global analysis of the entire project. It is shown how the Bayesian solution of

inverse problems for scientific inference can be considerably accelerated.

This thesis is an interdisciplinary work between the fields of physics and electronics. Given that both fields have different ways to discuss and present their results, this work tries to accommodate both. However, this may not have been successful in every aspect. This work is written and meant to be read as a whole for a proper understanding. Nevertheless, the results of section 3.4 and section 4.4 may be more appealing for the reader with a background in physics. The same applies to section 3.5 and section 4.5, where the results are focused on the acceleration and engineering aspects of the work.

## Other Publications of this work

The work has been presented at two conferences by myself and both papers were published in IEEEXplore proceedings. The dispersion interferometer model, design and acceleration discussed in chapter 3 was presented at the $13^{th}$ IEEE International Conference on Signal Processing in Chengdu, China [1].

The last part of this project, the joint analysis of the Dispersion interferometer and the Thomson scattering system, was presented at the $13^{th}$ IEEE Global Conference on Signal and Information Processing (GlobalSIP) in Quebec, Canada [2].

A general overview of the advantages of hardware acceleration of data analysis of plasma diagnostics and its comparison to a standard analysis is being prepared for publication in the journal Review of Scientific Instruments.

# 1. Introduction

## 1.1. Motivation: Data Analysis in Modern Scientific Research

In the field of data analysis for modern scientific experiments, one of the most common problems is that of parameter estimation. A scientific experiment is built to study, control or observe phenomena of interest. In order to record and study what has been observed in an experiment, a diagnostic is used to measure a specific parameter of interest. With those measurements, the possible behavior of the parameter of interest is analyzed to develop or validate a theory about the observed phenomena. This essentially describes the field of inverse problems, where the causal factors or parameters are inferred from a set of observations [3, 4].

Focusing on parameter estimation in inverse problems within modern scientific experiments, the data analysis faces a problem of a trade off between processing time and thoroughness of the analysis. In some cases, the parameters are measured having in mind control and safety of the machine in use, which means that approximated fast calculations are an advantage. In other instances, which is typically the goal of research experiments, the parameter is measured in order to perform scientific inference with a high level of accuracy. When inference is the goal, the need for a study typically means that the measured phenomena are not yet fully understood; and the introduction of approximations reduces the information content that we obtain from the parameter of interest.

Unfortunately, many studied parameters in modern experiments do not fall under only one of these categories. In many cases, like the plasma physics niche of this thesis, the unknown and studied parameters of interest are the same being used for control and safety of the experiment.

Increasingly used for different data analysis schemes, one of the currently most transcendent techniques in parameter estimation is Bayesian analysis [5, 6, 7, 8]. It provides a good way to deal with combinations of data from diagnostics measuring the similar parameters. Also, it allows for a rigorous handling of the uncertainty related to the parameter estimation. Last but not least, it simplifies the way to introduce our knowledge, or lack thereof, into the analysis. Like most statistical procedures, the processing requirements make it a common tool for a post-processing scheme.

Regarding the frame of this work, many plasma diagnostics are used exclusively to preform scientific inference on the physical process in order to estimate the relevant plasma parameters [7]. With other diagnostics, the data is also used for a real-time analysis that protects the experiment from reaching damaging conditions or to feedback control experimental parameters. This can occur in many ways like controlling the position of the plasma or turning off a heating

system [9].

As in many modern physics experiments, in plasma physics the evolution of plasma parameters and other processes tend to follow complex non-linear models with extensive analytic expressions. The time dynamics and variables of these models are also usually not known with certainty. Thus, the lack of information on dynamic behavior constrains the analysis, which is then normally performed for time-independent models where a single data point is considered.

This makes the frame of plasma physics an ideal field for standard Bayesian analysis where the introduction of approximations and linearizations is avoided in order to obtain unbiased information about the studied phenomena [8, 10]. Nevertheless, a Bayesian approach to these types of inverse problems is typically slow. It can have several data sources and free parameters where often the solution is intractable or has no standard analytic form.

The question, as the reader could have inferred by now, is how to deal with parameter estimation where unbiased Bayesian analysis is the best tool but the parameter measured is also required for real-time control. Can a faster version of this standard time-independent non-linear Bayesian analysis be achieved?

Since Bayesian analysis typically requires big amounts of processing time and power, it is clear that accelerations and optimizations of this procedure are needed in several fields where the same conditions apply [11]. An acceleration would serve not only as an unbiased tool for scientific inference but also for safe control systems with an adequate uncertainty estimation of the control signal.

Given the analysis' wide dispersal, several efforts towards this goal have been carried out and succeeded in bringing versions of Bayesian analysis to a real-time scheme. Bayesian filtering is a good example where various techniques have been sped-up for a real-time scheme. Most of these are used for analysis of time-dependent models with a state-space approach where filtering and smoothing are used for control systems [12].

Others focused on linear problems with Gaussian noise where the Kalman filter is the preferred tool and is in general a reformulation of Bayes' theorem [13].

In the case of non-linear problems of time-dependent models, a reduction in the volume of calculations can also be achieved with sub-optimal algorithms or linearizations within the Bayesian approach. Some examples are the extended Kalman filter (EKF), particle filters, grid based algorithms and variational Bayes techniques, amongst others. Most of these, through a Bayesian approach, target time-varying inverse problems and are formulated as a stochastic state-space model [14, 15, 16, 17, 18].

In a control theory frame, the majority of these are the best way to reach a real-time solution

in a dynamic model state-space approach. Nevertheless, they are not the most suitable solution for the requirements stated in the frame of this work.

Therefore, the following work's goal is to satisfy the need for a faster version of this analysis under the conditions that modern scientific experiments often present.

## 1.2. Fusion

Nuclear fusion is a promising alternative to solve the ongoing problem of an increase in energy demand. While far from being a simple subject, the basic idea behind fusion can be simply described as the reaction that happens when two nuclei have enough energy to overcome the repelling Coulomb force. Typically, fusion reactions produce a heavier nucleus and a neutron or proton.

The canonical example of fusion takes place in a star, where for most of its lifetime, it is powered by exothermic reactions that release energy. Here, the sum of the resulting particles in the reaction has a rest mass that is smaller than the sum of the original elements before the fusion process takes place. Through Einstein's $E = mc^2$ equivalence, a calculation of the energy released from that missing mass is possible and shows how the order of magnitude of the released energy grows towards the MeV range for a single fusion reaction.

For sustained fusion to happen in nature, it has to be in thermal equilibrium and have high thermal energy to overcome Coulomb forces. At these temperatures the particles form a plasma, which is a fully or partially ionized gas. This means that the confinement of a hot ionized gas which shows collective behavior provides the environment for the fusion of two light elements into a heavier one.

The reaction itself has a fuel energy density per unit mass higher than other typical reactions used for power generation. This means remaining optimization point is to increase the energy output by the proper selection of the fuel. For optimal results, the selection is done in terms of reaction rate and energy density per fuel mass. The heavy isotopes **Deuterium** (D) and **Tritium** (T) are considered the ones that best meet this requirement, specially the reaction rate. Their energy density per unit mass is reflected in table 1.1 which shows amount of usable energy per kilogram fuel. Also, the abundance of Deuterium in sea water and Tritium extracted from Lithium-6, removes the fuel availability limitation as a problem. Finally and in contrast to the fission process, the radiation half-life times of the isotopes used is short and can be dealt with through proper storage.

In order improve the conditions to achieve fusion for the two isotope nuclei, their kinetic energies must be high to beat the Coulomb repulsion. The likelihood of a collision is increased with confinement time and density as well. The relevance of these factors was initially seen by studying the power balance. The power balance can be constrained to a case where no external power is applied. If the power generated by the fusion reaction is transferred back to the plasma, it is considered self-heating. Under these conditions the "triple product" term shown in 1.1 can

Table 1.1.: Amount of Usable Energy per kilogram of fuel

| 1 kg | kW h |
|---|---|
| Coal | 8.1 |
| Oil | 11.6 |
| Gas | 13.9 |
| $U^{235}$ | $2.3 \times 10^7$ |
| D-T | $9.4 \times 10^7$ |

be derived. It is commonly used to define the balance required between temperature, density and confinement time to produce a positive power output.

$$n_e \tau_E T = 3 \times 10^{21}\,\mathrm{m^{-3}\,s\,keV} \tag{1.1}$$

Here the temperature T, is directly related to the kinetic energy required to overcome repulsion forces, the electron density $n_e$ relates to the probability for a collision of two nuclei and the energy confinement time $\tau_E$.

Achieving high kinetic energies, the energy confinement time and density depend strongly on the quality of the fuel confinement. In the case of stars, the massive fuel abundance plus the strong gravitational force provide the suitable environment for fusion to happen. When attempting to recreate this on earth, those two parameters have to be controlled through other means. For this magnetic confinements is used.

Magnetic confinement takes advantage of the behavior of charged particles in a magnetic field to constrain particle trajectory. Through magnetic tension and Lorentz force the outwards pressure that a volume of plasma will exert can be counteracted. The two most advanced types of magnetic confinement concepts are the tokamak and the stellarator, this work focuses on the latter.

### 1.2.1. The Stellarator

As mentioned above, satisfying the constraint of the three parameters in the triple product is not equally difficult. While reaching a required density and temperature to meet Lawson's criterion is not trivial, it is the confinement time that comes at a higher cost. Its control requires the ability to regulate the power losses in a plasma and needs a machine design that is able to properly confine it.

Given that charged particles follow a magnetic field, a good approach was to confine the plasma through a tailored magnetic field. The leading two approaches to build such a magnetic

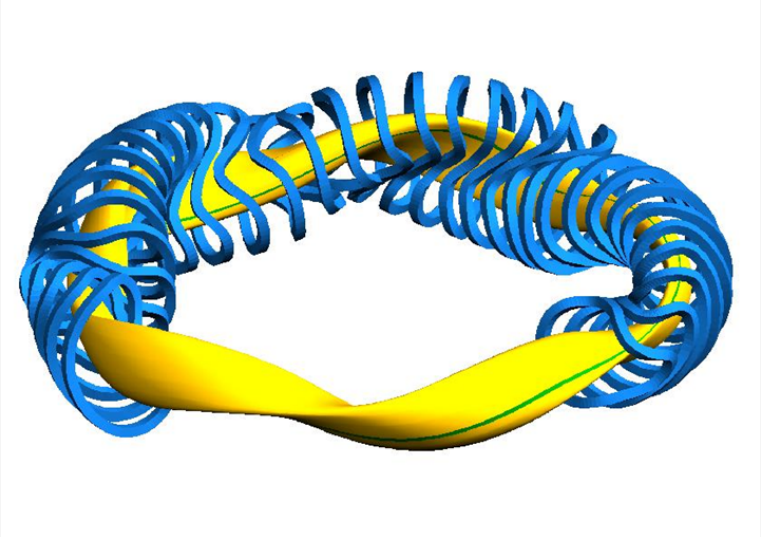


Figure 1.1.: Wendelstein 7-X modular coils (blue) configuration and the plasma (yellow).

confinement device were the mentioned stellarator and tokamak. The first idea for the stellarator design came from Lyman Spitzer Jr. in 1951 who thought of a way to solve the problem of confining a plasma [19] while avoiding particles drift out of the plasma due to several forces.

If a set of coils is arranged to form a torus, the resulting toroidally shaped magnetic field guides particles to travel around in a loop. Unfortunately, the effect of magnetic drifts makes the particles move radially outward and escape confinement. For the sake of this explanation the radius is measured from the center of the torus to the particle position. Lyman's alternative proposed a way to introduce a twist in the magnetic field. With this change, a particle will follow a magnetic line that no longer has a fixed radial position. A twist in the field would cause the lines, and therefore the particles, to constantly change their radial position so the drifts cancel out. These drifts can be caused by several factors including centrifugal forces as well as forces caused by the combination of the confining magnetic field and an electric field caused by charge gradients.

To achieve this, modern stellarators use modular coils that not only introduce a toroidal field component but a poloidal as well. The sum of both effects results in a twisted magnetic cage that provides a magnetic guide for the particle trajectory while reducing particle losses.

The shape of these coils and the resulting field can be seen in fig. 1.1. Here the coil design depicts that of the **Wendelstein 7-X** (W7-X) stellarator located at the Max Planck Institute for Plasma Physics.

While the plasma confinement is mainly dictated by the magnetic field, the control depends on many other parameters. Magnetic confinement devices require a set of peripherals that

provide control of the machine in different aspects; ranging from strength of the magnetic field to the amount of heating power deposited. The magnetic field contains a plasma with a temperature in the order of millions of degrees and is often heated with microwaves in the order of megawatts. Therefore, the measurements of plasma parameters through different diagnostics provide information for scientific inference and machine safety.

### 1.2.2. Plasma Parameters: Real-time monitoring and control

One of the central parameters in all fusion devices is the electron density. Proper measurement of the electron density and its uncertainty is of importance because of its fundamental role in plasma physics. It is involved in the determination of whether a burning plasma meets the Lawson criterion eq. (1.1). It also often used as a reference parameter for different heating configurations.

Through the local density estimation, parameters like plasma frequency, collisionality and plasma pressure can be estimated [20]. Besides this, its gradient is needed to estimate plasma transport. The knowledge of the density is also crucial to experimentally reconstruct the equilibrium magnetic field, which is perturbed by the plasma, and to predict macroscopic stability of these equilibria.

The investigations presented in this thesis are carried out within the frame of two plasma diagnostics that have density as a parameter in common. The Dispersion interferometer is selected for the initial part of the project due to its simpler forward model as a test-bench. This scenario as a proof of principle provides a good stepping stone into more complicated analysis like joint analysis with other Bayesian models integrating several parameters from different diagnostics. For the latter stage of the project, the **Thomson scattering** (TS) diagnostic was selected. Besides sharing a line of sight and a parameter with the interferometer, it has a model that better represents realistic complex examples of Bayesian analysis.

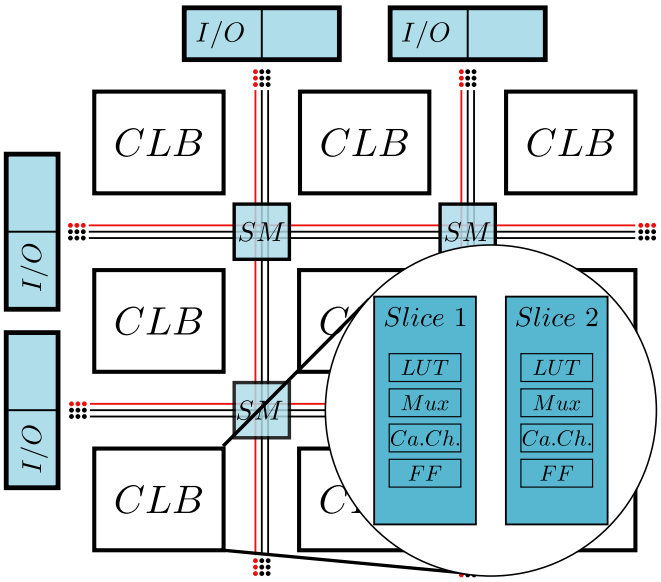



Figure 1.2.: General depiction of FPGA architecture. The resources included are: input/output blocks (I/O), configurable logic blocks (CLB), programmable interconnections, clock connections (red), switch matrices (SM). The CLBs consist of logic slices containing lookup tables (LUT), multiplexers (Mux), carry chains (Ca.Ch.) and flip-flops (FF).

## 1.3. Field Programmable Gate Arrays

A **field-programmable gate array** (FPGA) is a reconfigurable integrated circuit comprising arrays of logic gates and other resources, which allow the user to change their configuration by writing a code describes their interconnection.

Before their invention in the late 1980's by Steve Casselman, the option of changing a programmed circuit component was only possible with **programmable read-only memories** (PROM) and **programmable logic devices** (PLD) [21]. While they were reconfigurable individually, the connections between their components or gates was not, and a change in the design would normally require a long production time and cost.

Altera and Xilinx, two programmable devices companies, started producing and marketing options of this new reconfigurable technology after its invention. It wasn't until 1985, that the first device with reprogrammable gates and connections between them was released by Xilinx [22].

The architecture of modern FPGAs, similar to the initial Xilinx design, is in general composed of three main elements represented in fig. 1.2. They are **input/output** pads (I/Os) to interface with external signals, configurable logic blocks (CLB) to carry out different tasks and programmable interconnections between them. The interconnections can be reconfigured to

connect with different resources through transistor driven **switch matrices** (SM).

The CLBs are the basic unit in an FPGA and are composed of one or more logical slices (or cells). Depending on the model and manufacturers, the slices have different resources which allow the user to reconfigure the design to attend specific needs [23]. Within a slice, the resources available are a mixture of **lookup tables** (LUT), multiplexers, carry chains and flip-flops. Their combination allows for a wide variety of applications giving FPGA's their flexibility.

Among these applications is the implementation of arithmetic operations in different formats. Specifically for signal processing, modern FPGAs also have a different type of slice commonly referred to as **digital signal processing** (DSP) slice. The DSP slices have a different configuration that contain multipliers, pre-adders, accumulators, basic logic functions and pattern detectors amongst others [24]. While many of these more complex functional components (cores) can be built from normal slice resources (soft-core), the DSP slices have these specific cores already hardwired (hard-core).

With the combination of the previously mentioned resources, FPGAs provide a reconfigurable architecture that can be specified through a hardware description language.

### 1.3.1. Hardware Description Languages

The FPGA design is done with a branch of programming languages called **hardware description languages** (HDL). While they borrow many concepts from general purpose software programming languages, they differ in aspects that make HDL more suitable to describe hardware. One of these differences is that due to the hardware possibilities of implementing numerous parallel processes, the specification of time for each process is done explicitly. While in software programming languages it is possible to do multi-threading (running processes in parallel with multiple processors), their definition of when a process takes place is typically less deterministic. HDL is therefore used to design complex circuits in many fields from **application-specific integrated circuit** (ASIC) to microprocessors [25].

Several versions and standards of this language have been created. For FPGAs, the two mainly used are Verilog and VHDL. The work for this thesis was done with the latter which stands for **Very High Speed Integrated Circuit** (VHSIC) hardware description language.

These languages allow for the synthesis of a code into netlists, which specify components and how they are connected. In FPGA design, a netlist is typically routed, implemented and programmed on an FPGA. The simulation of the resulting design, or the intended circuit, can be done for several stages. These range from a behavioral simulation to a final simulation

considering the specific implemented hardware properties. For this, libraries containing tested circuit behavior and timing are implemented to show how the design should behave. This allows for a verification of the code before using it to program a reconfigurable hardware like the FPGA.

### 1.3.2. State of the art

Advances in FPGA performance, degree of parallelism and resource availability prompted their use in the field of high performance computing [26]. Comparing it to the use of normal computers, this field is characterized for its use of higher processing power and tailored optimizations for complex science and engineering problems.

Along with **graphics processing units** (GPU), FPGAs are the most common architecture used for acceleration of many well-known numerically intensive algorithms or applications. In some cases, FPGAs show promise in outperforming modern processors or considerably improve processing times [27, 28, 29].

The addition of arithmetic hard-cores or DSP slices to FPGAs is a relatively recent development that motivates the implementation of complex arithmetic functions and operations into an optimized real-time scenario.

Besides the inclusion of arithmetic hard-cores, FPGA manufacturers also offer **Intellectual Property Cores** (IP-cores). These IP-Cores provide the user with optimized hardware implemented versions of numerical algorithms, data transmission standards and many others. These cores can be configured depending on the required functional parameters and whether the optimization goal is area (resource usage), power consumption or speed. They can range from Ethernet layers and **Random Access Memory** (RAM) interfaces, to implementations of double precision floating-point arithmetic operations. The latter are particularly important for this work, because they decrease development time by reducing the developer's workload. By avoiding the re-design of existing optimized cores, the developer is able to focus on optimizations of the global design.

Finally, the resources in modern FPGAs allow the implementation of extensive arithmetic problems that a few years back, could have been impossible on a single chip. The work presented here uses both Virtex-6 and 7 families of Xilinx FPGAs where DSP slice and logic slice counts are considerably higher than several years ago. Other families like Ultrascale provide a higher resource availability at a considerably higher monetary cost and therefore are commonly used for massive designs or non-optimized research designs that must fit in an FPGA.

## 1.4. Outline of Contributions

The main contribution of this thesis was to prove that a general acceleration of Bayesian model based data analysis for inverse problems in scientific experiments, is feasible. This was done by demonstrating how modern FPGAs allow the acceleration Bayesian analysis of a range of possible cases in the field of inverse problems. This range of problems was covered by showing the feasibility of an acceleration in two representative cases and was carried out in several stages described below. Moreover, in this context acceleration is interpreted as the translation of a software code into a dedicated hardware design with the aim of reducing its processing time.

The first stage and representative case is the analysis of a dispersion interferometer, a diagnostic which has recently increased in popularity due to its advantages compared to other interferometers. It represents the cases of small models, non-linear models or multimodal posterior distributions. In addition, it addresses the use of an important Bayesian analysis method, the sequential approach, which is crucial for estimation of parameters with a longer dynamic time-scale than the available sampling rate. In this frame, the first contribution was the development of a Java code in the Minerva framework to carry out a Bayesian analysis for dispersion interferometer. This required finding a solution for a non-linear model which introduces ambiguity in the determination of the phase difference through a multimodal posterior distribution. For this, a code using sequential Bayesian analysis was developed. In order to define a reference point for posterior acceleration, a full implementation of this analysis was programmed, tested and validated in the Minerva framework.

This allowed for the second contribution, a 16-fold acceleration of this analysis. An FPGA version was developed, implemented and tested proving the acceleration possibilities of the analysis and promoting its use in a scheme different from the usual post-processing processing approach. The shown potential of FPGA acceleration of non-linear inverse problems motivated the second part of this thesis.

The second part is the other representative case, covering the acceleration of mathematically extensive and complex models, models with multidimensional posteriors and cases that require iterative sampling algorithms to carry out the inversion. Furthermore, it includes another powerful property of Bayesian analysis, which is the ability to do a joint analysis of models that have parameters in common. This was represented by the Thomson scattering diagnostic and its joint analysis with the dispersion interferometer. It resulted in a third contribution, the development of a Bayesian analysis code in the Minerva framework for the joint analysis of both diagnostics. As with the first code, the developed analysis code was tested against other

analysis methods and validated before its acceleration.

With a reference software code, the fourth contribution was proving that the Bayesian analysis of more complex models can be accelerated. Given the complexity of the analysis, a study of the most time-consuming sections was made in order determine where an FPGA acceleration would be more relevant. This was found to be the acceleration of the most time-consuming part of the analysis, the processing of the forward model. This design was tested and implemented on an FPGA in order to validate the results and the achieved 82-fold acceleration factor.

The complexity and size of the design required the development of a tool that simplifies circuit design and optimization of large mathematical models on hardware. This fifth contribution enables the developer to improve the design process by decreasing the development time and difficulty of going from an extensive mathematical model, to a full hardware implementation. It simplifies the application of pipelining and parallelism to reduce processing time.

Finally, by determining that it is feasible to accelerate these two representative cases with an FPGA design, this work establishes a precedent for bringing this type of Bayesian analysis out of a post-processing scheme. This is not only relevant for plasma physics experiments, as this work demonstrated. It is also applicable on any type of inverse problems where their analysis suffers from the introduction of approximations, yet a fast version is required.

# 2. Analysis Techniques

## 2.1. Forward Modeling

When dealing with the task of modeling a diagnostic, it is important to consider all the parts involved in the process of taking a measurement. The model must include the dependencies of the diagnostic on basic parameters, the physics involved in the process and the electronics involved which normally are all well-known.

Models of inverse problems often have the difficulty that a specific measured value can be explained by different combinations of its model parameters. Thus, the physics parameter estimation can become impossible. However, if the model description from parameters to an electronic measurement is well known, predicting a measured sample from a specific combination of parameters is possible and known as *forward modeling*.

Forward modeling presents an effective way to understand the relevance of the parameters in a final observation and properly represent diagnostic dependencies.

An example of this is shown in fig. 2.1 where the forward model of generic plasma density diagnostics is depicted.

The **forward model** (FM) has several advantages due to its clear way of showing the full functionality of a diagnostic. For instance, it allows correlating data coming from two diagnostics that have a parameter in common. Also, due to its capability of predicting a measured sample, synthetic data can be generated for studying the diagnostic's behavior without actual measurements.

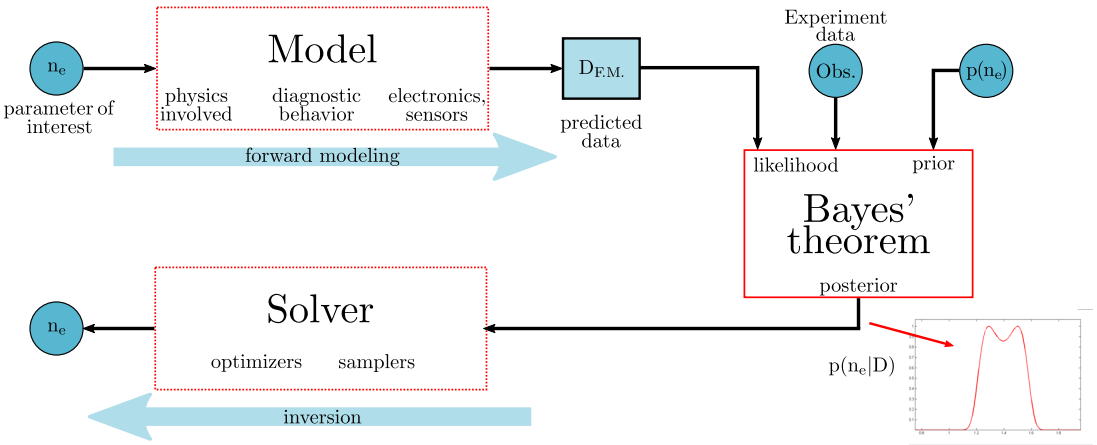



Figure 2.1.: Forward model, Bayes' theorem and inversion example for electron density ($n_e$) as main parameter.

## 2.2. Bayesian Analysis

### 2.2.1. The Likelihood Distribution

The likelihood distribution is a **probability density function** (PDF) that represents the probability of an observation being measured given the combination of the model parameters.

If our knowledge of every single intricacy of the model was absolute, and we could account for electronic noise and changes in a device's response to e.g. temperature changes, we could formulate the forward model as a function $f$ that yields a sample $\vec{D}$ given a set of parameters $\vec{p}$ so that $\vec{D} = f(\vec{p})$.

Since that is not the case, the likelihood distribution describes the uncertainties of the measurements in a PDF for all possible combination of parameters $P(D \mid \vec{p})$. It thus represents the probability of a observation happening for a chosen specific combination of parameters.

### 2.2.2. The Posterior Distribution and Bayes Theorem

Given that the only thing we can be certain of is the taken observation $\vec{D}$ and our interest is to know determine the distribution of $\vec{p}$, we must use Bayes' theorem to obtain the posterior PDF $P(\vec{p} \mid \vec{D})$.

Bayes' theorem states:

$$P(\vec{p} \mid \vec{D}) = \frac{P(\vec{D} \mid \vec{p})P(\vec{p})}{P(\vec{D})}. \tag{2.1}$$

Besides the posterior PDF and the likelihood when can identify two more factors. The first one, $P(\vec{p})$, is called the prior and encompasses the knowledge, or lack thereof, that we have regarding the state of the parameters of interest before the measurement is done. The second one is the evidence, $P(\vec{D})$, which is a distribution that depends only on the measured data. It is used to compare models but in the case of parameter estimation when using a single model, it is usually ignored.

The final posterior PDF that Bayes' theorem provides is a representation of all knowledge of the parameter's state when considering prior knowledge, the observations and the uncertainties associated. It provides a full description of our knowledge including correlations between parameters, amongst others.

### 2.2.3. The Joint Posterior Distribution

As it will be seen in the second part of this thesis, it is common practice and an advantage in Bayesian analysis to be able to do a single analysis for multiple diagnostics that share a parameter [30]. That is, two sources of data that measured the same parameter through different approaches. In such a case, a likelihood function will represent the forward model for each diagnostic and through probability theory a single posterior PDF can be expressed as

$$P(\vec{p} \mid \vec{D_1}, \vec{D_2}) \propto P(\vec{D_1}, \vec{D_2} \mid \vec{p}\,) P(\vec{p}). \tag{2.2}$$

Here the evidence factor has been removed and a proportionality is used when focusing on parameter estimation because the evidence term is constant when only one model is used. The previous is referred to as the joint posterior distribution and given that for a fixed parameter the noise is specific of each model, and they are independent measurements, it can be expressed as

$$P(\vec{p} \mid \vec{D_1}, \vec{D_2}) \propto P(\vec{D_1} \mid \vec{p}\,) P(\vec{D_2} \mid \vec{p}\,) P(\vec{p}\,). \tag{2.3}$$

### 2.2.4. Marginals and Conditionals

There are two final terms commonly used in Bayesian analysis that should be defined, the marginal and conditional distributions. These are useful when the interest lies in a specific parameter $p_1$ from the subset of parameters $\vec{p}$ used in the full analysis. The marginal and conditional provide a way to separate from the analysis those parameters that are necessary for the full analysis but not of particular interest. These irrelevant parameters are also referred to as nuisance parameters.

In the case of the conditional posterior distribution, as the name suggests, the posterior is evaluated for the parameters of interest when the conditions are fixed; that is when the rest of nuisance parameters have a specific value so that the posterior would be $P(p_1 \mid \vec{D_1}, p_2, p_3, \ldots, p_N)$.

On the other hand, the marginal posterior distribution permits the calculation of a specific parameter regardless of the state of the rest. For this an integration over each parameter is necessary where the marginal would take the form of

$$P\left(p_1 \mid \vec{D}\right) = \int_{-\infty}^{\infty} P\left(\vec{p} \mid \vec{D}\right) \, \mathrm{d}p_2 \, \mathrm{d}p_3 \, \ldots \, \mathrm{d}p_N \tag{2.4}$$

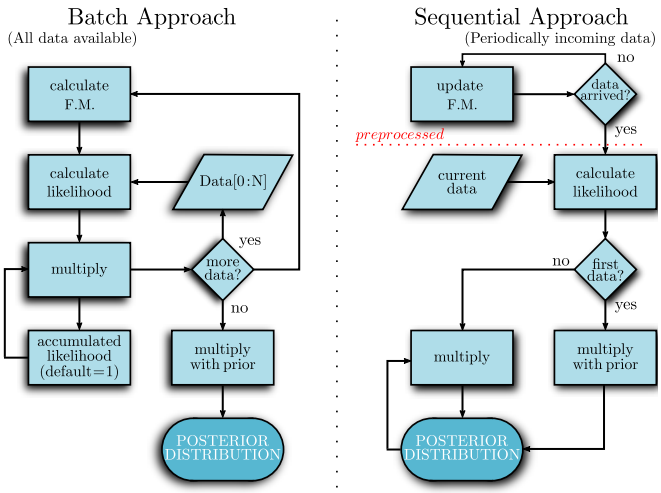



Figure 2.2.: Flowchart depicting the differences between a batch and sequential approach for Bayesian analysis.

For the cases of high dimensionality, the marginal is the only way to visually interpret the data and isolate the parameter of interest.

### 2.2.5. Sequential Bayesian Analysis

The availability of data for the analysis can vary from one problem to the other. In some cases a batch of data is available at once while in others the data comes sequentially (i.e. samples received as they are taken).

In the cases where data could be analyzed as it arrives, while the complete set of data is needed for the analysis, a sequential treatment can solve the problem. Considering that each sample is independent of the next one, we can reformulate Bayes' theorem in a sequential form as:

$$P(\vec{p} \mid D_1, D_2, \ldots, D_N) = P(D_1 \mid \vec{p})P(D_2 \mid \vec{p}) \ldots P(D_N \mid \vec{p})P(\vec{p}). \tag{2.5}$$

In the frame of this work where acceleration is the main objective, this approach is used in order to calculate a posterior of a single sample which can then be used as a prior for the following to update the posterior.

### 2.2.6. Inversion Algorithms

Depending on the nature of the problem, its linearity, number of modes and dimensionality, different inversion techniques are available and define time and processing power requirements. The inversion, opposite to the forward modeling, is the search for the parameters $\vec{p}$ that yield

the highest probability density on the posterior. This specific combination of parameters $\vec{p}$ is called the **maximum a posteriori** (MAP) and requires the finding of the maximum value of the posterior distribution.

The modeled system can have specific characteristics that require a different sampler algorithm or optimizer tool to estimate parameters and their uncertainties.

The linearity of the model, the number of modes or local maxima in the posterior distribution, and its dimensionality changes the applicability of inversion algorithms for estimating the MAP. Linear problems can be approached by **Linear Gaussian Inversion** (LGI) while non-linear ones might lead to long iterations with a sampling algorithm [31].

Depending on the mentioned factors, different inversion algorithms are chosen to decrease the processing time and power required.

### 2.2.7. Minerva

The work described in this thesis uses Bayesian inference for diagnostics based on the software framework **Minerva** [32]. As a framework, Minerva provides the tools required for Bayesian analysis through Bayesian graphical models. It also provides a modularized scheme that allows the decoupling of models, parameters and inversion algorithms. That way a model can be easily tested with different optimizers or samplers without much change in the code.

The use of graphical models and the way it is built means that Minerva is very modular. This is one of the reasons why reconfigurable hardware like the FPGA is selected for acceleration in this work. The possible changes in models and inversion techniques benefit from an implementation platform with a modularity that allows the simple exchange of elements in the analysis. For this reason the analysis gains from the modularity and element swapping that Minerva and the FPGA provide.

### 2.2.8. Bayesian Theory and Terminology in this thesis

Bayesian analysis has spread across many fields, as a result different nomenclatures have been adopted. Due to the fact that the reader might be familiar with other forms of Bayesian analysis, like the ones presented in section 1.1, some terminology for the frame of this work must be defined to avoid confusion.

The concept “sample” can be interpreted as measurements or observations taken in an experiment as well as the ones taken by a sampler algorithm. For this work the former is the main use of the term and sample will be the observation measured during an experiment unless

specified otherwise.

When referring to Bayesian analysis, the term has also become broad enough that it encompasses different techniques specific to the field where they are applied. Therefore, the use of the term "Bayesian analysis" in this thesis strictly refers to the explicit usage of Bayes' theorem for inferring parameters from measurements in scientific experiments where the underlying phenomena is not completely understood.

## 2.3. Non-Linear Algorithms

Non linear cases require different techniques and algorithms to solve the inversion problem. Two main approaches used are: To find a maximum in a single-mode posterior with low dimensionality by the use of optimizers, or to sample the distribution with the use of sampler algorithms for higher dimensionality. In the frame of this work, some of the most relevant algorithms are presented because they were either used on the analyzed models or provide a solution to problems that are not covered in this work.

### 2.3.1. Gradient Descent

For situations where the posterior distribution has a low dimensionality and is smooth enough with a single maximum, the use of gradient search or other line search algorithms can be an appropriate technique. Working in the parameter space, the algorithm chooses a direction and searches for the optimum position in that direction. Gradient search involves the estimation of a gradient to choose the next step in a function that defined and differentiable around the current position vicinity. Complex calculations of the posterior can become computationally expensive for this method, since the step size must be chosen carefully and the number of operations given by the gradient calculations does have a big impact on processing time. Accelerations of this algorithm have been developed providing a possibility to implement fast inversions [33].

### 2.3.2. Pattern Search

Pattern search methods are direct search algorithms that do not require the estimation of a gradient to find maxima or minima. This property allows them to be used on discontinuous functions. The method works by changing a parameter by steps of the same magnitude until no increase or decrease can be reached. In some variations, the step size is halved and the method is reapplied. An example of pattern search methods is Hook and Jeeves, which is useful for simpler cases of low dimensionality. While practical and fast, this method still has problems with multimodality and only provides the MAP [34, 35].

### 2.3.3. The Metropolis Hastings Markov Chain Monte Carlo Sampler

The Metropolis Hastings **Markov Chain Monte Carlo** (MCMC) is a sampler method that allows taking representative samples of the distribution in order to either study the posterior with the taken samples or to generate a histogram to determine the maximum value [36]. One of its limitations is the requirement of knowledge regarding where the maximum could lie in

order to reduce convergence time. This means that in order to keep the number of iterations low, an optimizer is usually applied to have a good guess for the maximum. If these conditions are not met, the MCMC can require extensive sampling iterations and excessively sample areas far away from the maximum, therefore requiring long processing times.

MCMC takes samples by doing a random walk along the posterior distribution. Every "step" or change in position is sampled from a smoothed or trial distribution. A comparison of the next state with the current jump is made to decide whether the position will be changed or not. If the probability of the current state is higher than the new one, a new jump is calculated disregarding the current one. On the other hand, if the probability of the current state is lower than the proposed one, the current state is updated and the algorithm proceeds from this new position. This is defined as a chain and, if it is allowed to sample randomly for a long enough time, representative samples of the distribution around the maximum can be taken. Part of the complexity of this method lies in the determination of the trial distribution. This can affect the method's behavior by increasing the time it requires taking representative samples or by rejecting jumps more often. A common solution, as well as the one used in this work, is to shape the trial distribution to be proportional to the covariance matrix. The matrix holds information about parameter uncertainties and has a similar shape as the posterior.

Some MCMC implementations have the disadvantage of getting trapped in local maxima on multimodal posterior distributions. In this case, other versions that solve this problem by introducing parallel sampling chains exist.

### Parallel Tempering MCMC

**Parallel tempering MCMC** (PT-MCMC) is a method that uses several tempered chains to improve the dynamic behavior by helping the MCMC method in two ways [37]. First, it avoids the entrapment of a chain in a local maximum of a multimodal distribution and second it helps the main chain collect representative samples faster by improving the chain mixing. With PT-MCMC, each chain samples from a different trial distribution that is estimated according to the temperature of the chain. In general terms, the temperature of a chain is represents the likelihood that the chain has of sampling from a low density regions of the distribution. The higher the temperature the wider the phase space it can sample. The first and coldest chain will sample from the target distribution while the other hotter chains will sample from modified versions of the target distribution which are easier to sample. This makes the hotter chains move faster along the whole distribution and colder chains more likely to get trapped in a local maximum. After a defined period of iterations, samples are exchanged from a chain to a colder

one in order to sample effectively the whole distribution.

Optimized versions of this algorithm have been recently developed and a particular version using the parallel capabilities of the FPGA is of special interest to this work [38, 39, 40]. Given that the focus is to reduce processing time, an FPGA version of MCMC that can get closer to the maximum in a shorter time is a big advantage. Also, the fact that this method is less susceptible to getting trapped in a local maximum makes it a better suited alternative for a general acceleration solution.

## 2.4. Optimizing the Implementation of Extensive Arithmetic Functions in Hardware Architecture

### 2.4.1. Design aspects: Implementing arithmetic models on hardware

This project involves the design and implementation of arithmetically complicated functions with large number of mathematical operations that will be implemented in hardware. For this to be efficient, this work merited developing a tool to reduce the design and development time. The developed tool simplifies the hardware design of the arithmetic functions in realistic inverse problems and shortens the development time scale.

When implementing complex and extensive arithmetic functions in hardware, one of the most time-consuming and challenging tasks is the mapping of these functions into operations that can be optimally implemented in hardware. This optimization is done to properly map the function into a combination of available hardware operations. This can be targeted to optimize the critical path to reduce processing time or to reduce the number of resources required to implement such a function.

For the latter part of this work, FPGA design is compared with computer calculations in scientific inference. In software, these calculations are typically done with double-precision floating-point operations. In order to match this precision for a reasonable comparison, the FPGA design requires using either fixed-point arithmetics with long word lengths (number of bits used to represent a value) or double-precision floating-point arithmetics. Their use introduces either a longer latency factor or a bigger consumption of resources which adds another level of complexity for optimization. When trying to accelerate an analysis, this optimization is done to achieve a better critical path (maximal delay path between data input and result output) while staying within the use of available resources.

Another important hardware implementation concept is the increase of a circuit's throughput at the cost of operation latency, called **pipelining**. This method divides a circuit or operation into individual processing elements in order to process multiple data simultaneously in a single circuit or "pipeline" [41]. This normally increases the throughput, the amount of data that is processed per unit time i.e. how often a value exits the pipeline. Because each stages of the pipeline are connected in series, the latency of an instruction is generally increased given that each stage must wait the time the slowest stage requires. When pipelining hardware implementations or an arithmetic model, the timing of each operation has to be carefully considered. Values must meet the pipeline order and reach the next operation node at the same

time as its counterpart value, coming from another parallel thread.

Finally, one must consider that these extensive arithmetic models can be factorized or rearranged in many ways that could eventually result either in a shorter critical path or a smaller amount of operations and hardware resources. This makes hardware optimization of extensive arithmetic models a cumbersome, time-consuming task; regardless of whether the goal is critical path reduction or smaller resource consumption. FPGA design often requires both of these elements to be optimized and directly affects its development time.

### 2.4.2. Improving the work-flow and simplifying the design optimization process

To address the described difficulties, a Python package was developed to improve the visualization of the relevant hardware implementation aspects. The main goal was to decrease the time that is required to go from a set of equations that describe the model to a final optimized hardware arithmetic design.

The program does this by parsing the required equations and delivering a tree-like graph representing the circuit structure. This allows to visualize the important previously mentioned factors: Individual hardware operation latency, arithmetic terms dependencies, critical path length, the timing of pipelined parallelized threads and the effects of different factorization possibilities. With it, the developer can simplify the optimization of the function implementation in hardware since often extensive models can often have dozens of configurations that favor different optimization aspects.

#### Preparing the arithmetic model

Before using the tool to visualize the possible hardware implementation and start the optimization, the model has to be worked on due to several reasons.

The first reason and limitation is the availability of arithmetic operations that can be implemented on the selected hardware implementation platform. Given that this tool was developed for and used on the latter part of this project, the platform is an FPGA and the available operations are the double-precision floating-point arithmetic cores. The function is therefore rewritten to suit the available IP-Core library.

The second reason is the fact that each model or equation can be factorized or manipulated in different ways that favor either the number or the type of operations required. For example, physics equations can sometimes be algebraically manipulated in order to express them in terms

of one relevant variable. This does not necessarily coincide with the factorization that would contain the least number of operations or the least number of divisions, which is generally preferred in hardware implementations.

These different formulations of the model's equation can be obtained manually or with a symbolic mathematical equations package in Matlab, Python or even the WolframAlpha computation engine. With it, several versions of the re-factorized function can be analyzed to favor the selected optimization aspect.

Finally, if the implemented model is not extensive with numerous terms, it can be implemented as is. If not, it can split into a number of terms or groups of terms for individual analysis as $A_2$ in eq. (2.7).

#### Using the tool to analyze possible optimizations

Once the right mathematical expression for the model has been prepared, it can be written in the tool as the input. The tool will use this input to create a tree structure where the output of the analysis would be the **root node** and the inputs would be the **leaf nodes** or end nodes. For this particular use of the tree structure, the length between a node and its parent is defined by the number of cycles that the hardware implementation requires to calculate the operation. This means that each parent node will be an arithmetic operation of the equation between two or more child nodes (operands).

Starting from the leaf nodes, the constants are available from the start of the analysis and just like the inputs, have no need to be processed. This means they also represent a leaf node and must be recognized by the tool. The tool is able to recognize numerical constants in the input equation but needs to be notified of the alphanumerical constants. This is done by initially filling a Python list with all the known constants present in the equations, as seen with $K_1$ in 2.6. In some cases for a specific analysis purpose, it is desirable to ignore the branch of a function. These functions are also added to this list so that the tool treats the branch as a single value or constant and ignores its operands from the final graph. Their specification is done the same way the constants are defined, as *sinTheta* does in

$$\mathrm{R} = [[\text{“K1”}, 0], [\text{“K2”}, 2], \text{“sinTheta”}]. \tag{2.6}$$

Since these functions will not be ready at time zero like constants, their latency or clock cycles required for its calculation can be specified as K2 shows in eq. (2.6). In this case a duration of 2 clock cycles is indicated and it will be placed in the respective level in the tree.

With the constants specified, the equations to be implemented are passed to the tool as a single equation or a set of them. In eq. (2.7), $A_{0,1,2}$ show the input as a set of equations while $B_0$ shows the equivalent as a single input.

$$\begin{aligned} A_0 &= 4 + 30\zeta^2 - 55\zeta^4, \\ A_1 &= 25 + \zeta^3(29 - 42\zeta^2), \\ A_2 &= A_0/A_1, \\ B_0 &= (4 + 30\zeta^2 - 55\zeta^4)/(25 + \zeta^3(29 - 42\zeta^2)), \end{aligned} \tag{2.7}$$

The set of equations has the advantage that it lets the user define the name of the parent node in the tree visualization. In the case of a single input, each operation will get the name of the list index where its stored with an R (register) prefix.

The final input required from the user is the latency of the functions for their hardware implementation and the definition of special functions. The definition of a latency allows the tool to properly display the levels of the nodes and distances between them. The special functions are defined due to the cases where a particular operation, like multiply and add, is available on hardware as a single operation with three inputs.

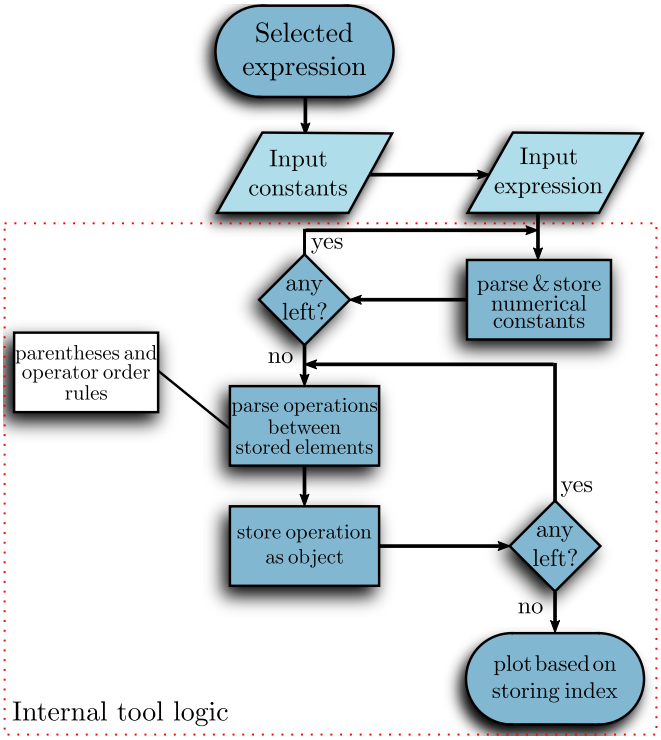



Figure 2.3.: Flowchart representing the use and internal logic of the optimization tool.

**Description of the tool's mode of operation**

The tool's work-flow and mode of operation, described in fig. 2.3 is the following: The first task the tool does is to parse each input line to identify constants, variables and operators. This is done procedurally to preserve the order in which the operations are made, considering parenthesis and arithmetic rules.

If there are numerical constants, it stores them as objects on the register list. The next step is parsing and storing operations that depend on previously registered objects on the list. Subsequently, the parsed operation becomes an object itself and is appended to the list. Thus, it becomes available for the following parsing iteration. This procedure is repeated until all operations on the expression are parsed. The program continues parsing each line with a new expression in the same way while respecting the hierarchy.

After all the expressions have been parsed, the total arithmetic function is stored in the list with their respective hierarchy, operation latency and dependencies. The tool is then ready to plot visual representation of the tree containing all the function's relevant factors in a hardware implementation as show in fig. 2.4.

Each parent node will be plotted with a name, or index number, and the operation symbol that it represents. The line length from each node to its parent node is proportional to the clock cycles required for the other sibling node to be ready. This indicates graphically how long a signal must be buffered and shifted for it to match its pipelined counterpart. The latency initially defined for each hardware operator is thus visually represented and reflects the real timing paths of each operation and its child branches.

To simplify the timing analysis, markers are placed at clock cycles intervals defined by the user. In the previous example the interval is 5, making the top branch 30 clock cycles in duration and the square root latency 10 clock cycles. This speeds the timing analysis when introducing shift registers to correct timing differences from the operand's counterparts in other branches. This can be seen for the multiplication of the unnamed registers with index $R17$ and $R11$ on fig. 2.4. Here, the $R11$ must be buffered and shifted 5 clock cycles until $R17$ becomes available.

Finally, if a factor is shared in different branches, it becomes available in the other branches at the tree level where the node that represents its calculation is plotted. For simplification of the visual representation, the package stores and displays each factor branch once. If the factor's branch is shared by several functions, its name or register index is displayed instead. A work-flow example of the tool can be found in appendix A.1, where the example depicted in

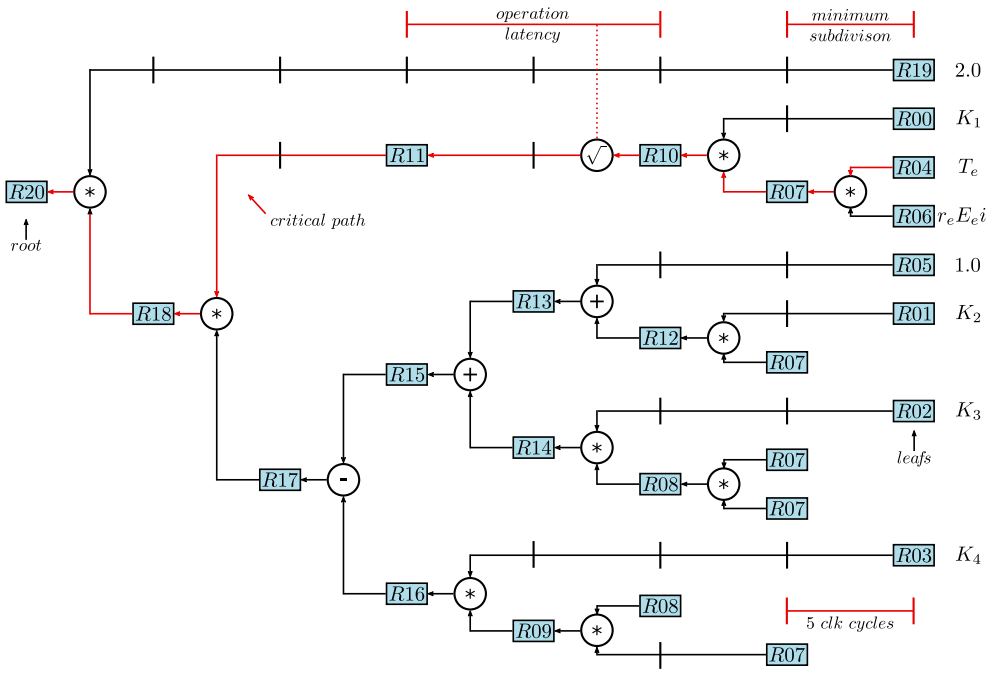



Figure 2.4.: Graph of tool output with registers (squares) and operations (circles). Minimal subdivision: 5 clock cycles. Original tool output can be found in appendix A.1

.

fig. 2.4 is thoroughly explained.

When analyzing the effectiveness of the tool in general, automated visual representation speeds up and favors the design process in several ways. As previously mentioned, a function can be algebraically manipulated to be represented with different expressions. This tool allows for a fast input of these different expressions to compare their critical path length, dependencies buffering and the number of operations needed. This is particularly useful for the implementation of extensive arithmetic models as it often is in scientific research. Depending on whether the optimization is regarding area, power or resource usage, the best expression for the implemented function is chosen.

Besides the optimization advantages considered, the tool also helps with implementation and coding aspects. It trivializes the calculation of operand buffering to meet timing of its pipelined counterpart. This is also advantageous when the different operation latency are tested to improve the critical path. With the use of this tool only a change of an input parameter is required.

In general, its use when implementing extensive arithmetic functions in hardware significantly reduces the development and optimization time, and simplifies the circuit analysis. It becomes good aid in the work-flow before and after the coding of a function in hardware architecture like the one implemented in chapter 4.

# 3. Acceleration of Interferometry Data Analysis

Interferometry is the technique of measuring interference on the addition of waves, typically electromagnetic waves. It is generally carried out by allowing two waves that have traveled through different mediums or conditions, to interfere constructively or destructively depending on the phase difference imposed on them by the path. From the resulting interference pattern it is possible to infer information about the medium that the waves went through.

From testing quality of reflective surfaces to measuring long and small distances, interferometry has a wide variety of applications. Here, we focus on the measurement of a refractive index. While there are many types of interferometers, this work uses the **dispersion interferometer** (DI). Besides being the available design for this work, it is a newer type of diagnostic that outperforms similar types. Its advantage is the use of one single optical path for both probing components which reduces the phase estimation to a single one while being insensitive to noise introduced by mechanical vibrations in the optical components. The following chapter will focus on the description of the Dispersion interferometer operating principle.

In order to establish a context for the modeled physics principles, the approach initially guided by its operation principle and design. The diagnostic's process to obtain a measurement will be described initially. Subsequently, an explanation of how the active parts of the diagnostic interact with the plasma will be used to associate how the physics phenomena occur and how it's modeled.

## 3.1. W7-X Dispersion Interferometer

As mentioned, one of the many uses of an interferometer is to measure a refractive index. The refractive index is a number that describes how light propagates through a medium as opposed to its propagation in vacuum. This means that in any other medium than vacuum, the group velocity of light will be lower. If one compares the phase of two light waves going through a medium and vacuum respectively, light traveling through a such medium will perceive a phase retardation compared to its vacuum counterpart. This phase shift carries information about the medium of interest and is the basic application of interferometry in this thesis.

### 3.1.1. Design and Operation of the W7-X DI

The Dispersion interferometer diagnostic is used to measure the plasma refractive index. The plasma refractive index is a function of its density and thus a measurement of its index allow us to estimate the line integrated electron density. The interferometer can be viewed as an active diagnostic that introduces negligible perturbations in the plasma by sending an electromagnetic wave through it. In this context, an active diagnostic is one that probes the plasma while performing a measurement. A passive diagnostic would be one that measures it without perturbing it in any way e.g., measuring the radiation of a plasma.

As with other interferometers, the basic principle can be split in three basic aspects. The first one is the generation of two probing components or beams in order to compare their phases after crossing a medium. This can be done with a reference line that doesn't cross the medium or by sending both beams of two different wavelengths through the medium.

The second aspect is to drive the beam with a set of optical components through the medium to be studied. In the case of a reference beam, only one probing beam will be sent through the medium and the other will travel the exact same distance without crossing the medium. In the case of two wavelengths, both are sent through the medium. An important point of this stage is that changes in the path length due to vibrations for example, can induce errors in the phase measurement.

Finally, the last aspect is to collect an interference pattern given by the constructive or destructive addition of both beams reaching a detector with their respective phases. In this chapter, the DI described will be the one built for W7-X [42, 43].

For the DI, the first aspect of generating the probing beams is done differently since both components travel the same exact path. Therefore, the diagnostic's operation starts with a single 20 W continuous wave $CO_2$ laser source operating at a 10.6 µm. It is directly shot to a

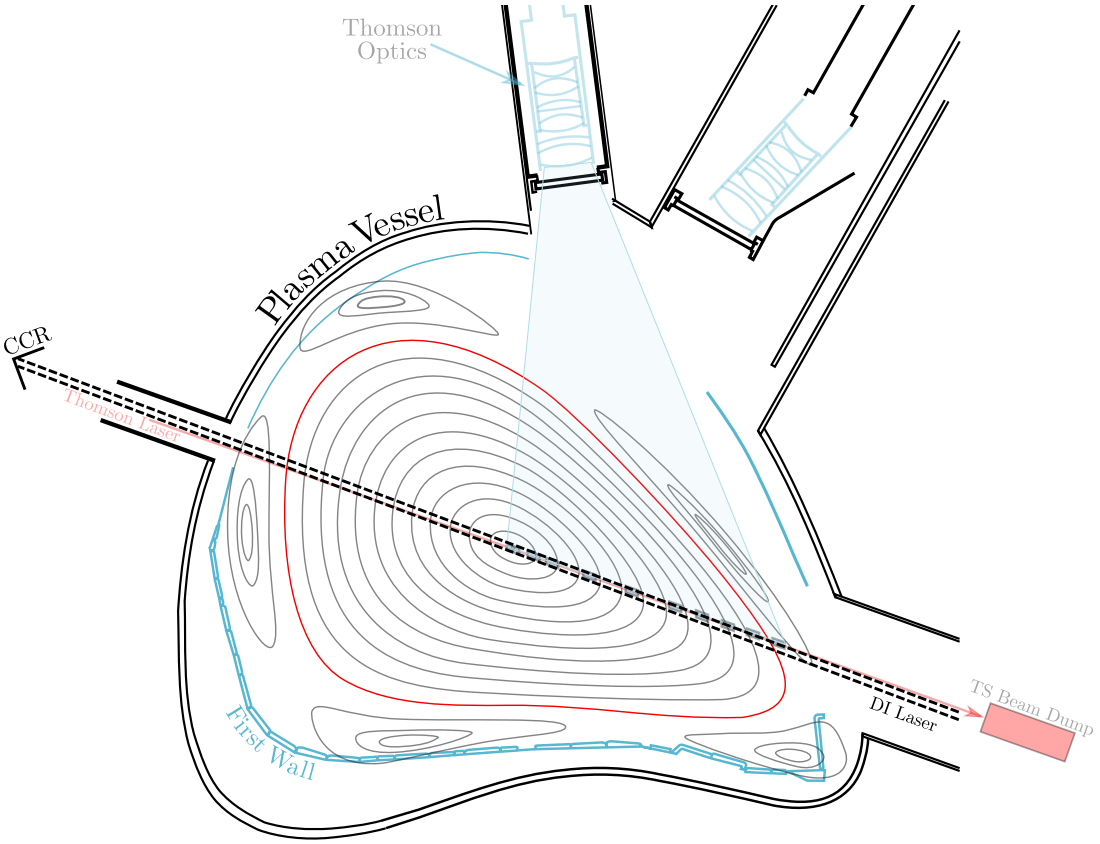



Figure 3.1.: Dispersion interferometer cross-section showing laser paths (dotted black), retroreflector (CCR) and plasma magnetic surfaces (concentric paths).

mirror that guides it through an attenuator used to dim the laser power for calibration and testing scenarios. After the attenuator comes another mirror which belongs to a group of optics that will not be mentioned since they are mainly used to redirect the laser to the main active components. They are nevertheless depicted in fig. 3.2.

The aforementioned mirror drives the laser to a telescope with **ZnSe** lenses that focuses the beam in a 5 mm × 5 mm aperture, $AgGaSe_2$ **frequency doubling crystal** (FDC) where the two probing components will be created. This crystal takes the first harmonic of the incoming beam, introduces a 90° polarization and doubles the frequency of a percentage of the incoming first harmonic power. Now both components have different frequency and travel the exact same optical path. This reduces noise due to changes in the path length of each component.

The doubling and polarizing efficiency, which is measured around $1.41 \times 10^{-5}$ %, sets the constraint for laser power required in order to measure a strong enough signal. For the final laser light to be properly measured, the laser power has to be in the order of the chosen 20 W laser capability.

Once the probing components are prepared, they have to be properly guided to the plasma

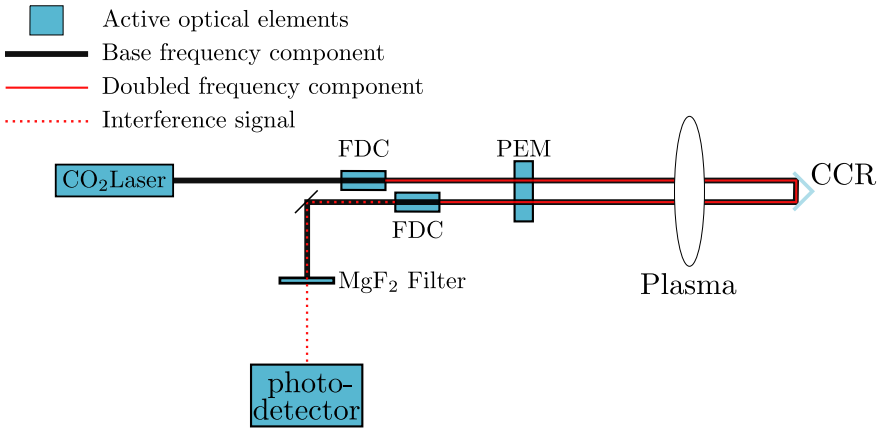



Figure 3.2.: Component placement and operating principle of the DI: $CO_2$ laser, frequency doubling crystal (FDC), photoelastic modulator (PEM), corner cube retroreflector (CCR), $MgF_2$ filter and the photodetector.

and back. After the FDC, the laser is sent through a ZnSe **photoelastic modulator** (PEM) which introduces a phase modulation to the polarized 5.3 µm component. This modulation is introduced to simplify final phase difference estimation according to a heterodyne scheme and it is applied at a 50 kHz frequency. It will be discussed further in section 3.2 when analyzed with the model for the interferometer.

Applying this modulation to only one of the components requires that the PEM only affects the modulation in one direction. It is therefore placed so that it coincides with the 90° polarized component without affecting the base frequency.

These two coaxial beams are sent to a concave mirror that guides them to exit the interferometry plate. This plate, the main support structure of the diagnostic, is a 2x2.50 m vertically placed basalt plate with another basalt neck of 2.3 m that positions the beam closer to the entry point in the torus.

The entry point to the Torus is a ZnSe window that the outer vessel has where both beams are focused to cross the plasma twice by having a **corner cube retroreflector** (CCR) located on the opposite inner side of the torus. This retroreflector sends the beam on an inverse trajectory and displaces it 5 times its beam waist radius.

The double-crossing of the plasma represents, as it is explained in further detail on the following chapter, that the plasma refractive index will have twice the incidence on the phase value than both beams have before entering the vessel. Also, that given that the frequency is different, both will experience a dissimilar phase shift.

Once the crossing of the medium is done, the generation of an interference pattern is the final aspect. On the return path for both beams, they are directed one last time into a second

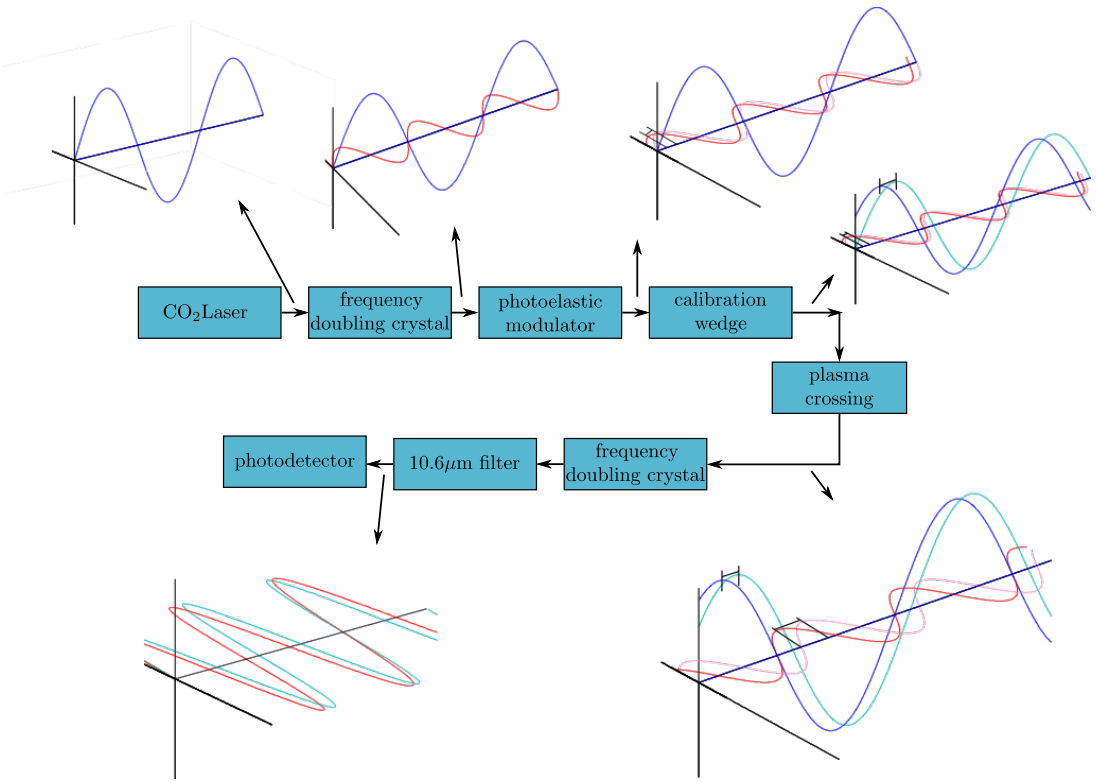



Figure 3.3.: Different stages after each DI optical active elements from the point of view of wave components

frequency doubling crystal. This crystal has no effect on the already perpendicularly polarized 5.3 µm component, yet polarizes and doubles the frequency of the main 10.6 µm component which has now experienced a phase shift from the plasma as seen in fig. 3.3. After this crystal, an interference pattern between the two components that crossed the plasma at different frequencies is created. This interference pattern will be the carrier of information about the plasma refractive index. The rest of the original first harmonic frequency will then be filtered out by a $MgF_2$ filter that is placed in front of the photodetector recording the signal with an **analog to digital converter** (ADC).

## 3.2. The Dispersion Interferometer Model

When modeling an interferometer in the field of plasma diagnostics, the physics principles required are that related to the measurement of the refractive index of a material, which in this case is represented by the plasma. For fusion devices and plasma refractive index measurements, there are several interferometers like the two-color interferometer or the Michelson interferometer that have been used to determine the line integrated electron density [44].

This chapter focuses on the newer concept of the dispersion interferometer which has also been developed for several fusion devices [45, 46, 47, 48].

When the beam path length is known, a refractive index can be estimated and vice versa. This index can be measured when the conditions are such that the medium is partially transparent to an electromagnetic wave. In a plasma, there are three main scenarios for the propagation of an electromagnetic wave. The first scenario is the total reflection of the wave which can be analyzed by comparison with reflection in a metal. In a metal the electrons are relatively free to move in the conductive band, shielding the propagation of the incident wave. They absorb the incident energy inducing a current that, similar to an antenna, will re-emit the wave causing a near total reflection. In the plasma, this can be deduced through the plasma frequency $\omega_{\mathrm{p}}$, which can be seen as a measure of how fast electrons can react to a charge displacement and thus how they can shield an electric field. It depends mainly on the electron density and is defined as

$$\omega_{\mathrm{p}} = \sqrt{\frac{n_e e^2}{\epsilon_0 m_e}} \tag{3.1}$$

where $n_e$ is the electron density, $e$ is the elementary charge, $\epsilon_o$ is the dielectric constant of vacuum and $m_e$ is the electron mass. This means that if the density is high and the plasma frequency is higher than the laser probing frequency $\omega_{\mathrm{l}}$, the plasma behaves similar to a metal with free electrons and can totally reflect the incident wave. This introduces the first limitation of $\omega_{\mathrm{l}} > \omega_{\mathrm{p}}$ for the probing beam to propagate.

The second scenario is a strong refraction of the incident wave. Similar to light refraction when the incident wave changes media, the plasma frequency can also affect the refraction index of the medium. Given that the plasma is not a homogeneous medium, its refraction index will depend on the local conditions of the plasma and is also a function of the plasma frequency and the probing frequency. If both frequencies have similar values, the probing beam will experience variable refraction due to density gradients along its path. These changes in the refraction index will cause the beam to change its geometrical path and miss the desired optical components.

This introduces a second design parameter that involves selecting a probing frequency well above the plasma frequency.

This brings us to the third case; full propagation to the medium. Here, given that the wave does propagate, the selection is done regarding how sensitive the phase change on the wave is with respect to the plasma refractive index. The refractive index is defined as the speed with which a wave propagates through a medium with respect to its speed in vacuum. This means that the lower the refractive index the bigger the phase shift experienced by the wave after crossing the medium. A lower frequency will translate to a higher sensitivity and bigger phase change due to variations in the refractive index with the risk of suffering too much refraction. A higher frequency means a lower effect of the density on the phase shift and thus loss of sensitivity i.e. the plasma is almost transparent to the wave.

With all this knowledge, a selection of the probing frequency can be done using W7-X design parameters and constraining the phase shift to $2\pi$ rad, thus avoiding phase jumps which can induce errors in the estimation.

Given that the W7-X heating system has an effective heating density limit, one can estimate a limit of maximal electron density or cutoff density of $n_c \approx 2.4 \times 10^{20}\,\mathrm{m}^{-3}$ [49] and find a limit plasma frequency of 0.8 GHz. This allows us to treat the plasma as a dielectric medium where, even considering the limit case, the selected laser wavelength satisfies the condition with $\lambda_l = 10.6\,\mu\mathrm{m}$ with a frequency of 177.7 THz [50]. With this selection, a change in density of $1.56 \times 10^{20}\,\mathrm{m}^{-3}$ would produce a phase shift of 4.66 rad and meets the design constrains.

Once the dependency to the refractive index of the plasma has been defined, we can proceed to use the expression for the relation between a phase difference $\Delta\varphi$ and a density value. This can be modeled as

$$\Delta\varphi = (2\pi/\lambda) \int_{l_1}^{l_2} [\mu_v - \mu_0(l)]\,\mathrm{d}l \tag{3.2}$$

where $l_2 - l_1$ is the path length, $\mu_0(l)$ the plasma refractive index and $\mu_v = 1$ the vacuum refractive index. Expressing the plasma refractive index in terms of density we obtain the expression

$$\Delta\phi = \left(\frac{\lambda e^2}{4\pi c^2 \varepsilon_0 m_e}\right) \int n_e(l) dl = \left(\frac{\lambda}{2\pi c} c_p\right) \int n_e(l)\,\mathrm{d}l, \tag{3.3}$$

that describes how the line integrated electron density is proportional to the difference between both phase components.

The final step is to model the diagnostic itself and how the measured signal intensity relates

to this phase difference.

In this case we need to define phase shifts that occur in the dispersion interferometer. The first are the shifts in the optical path length $\Delta d$, due to mechanical vibrations in the optics $\omega_l \Delta d/c$ and $2\omega_l \Delta d/c$ for the second harmonic where $\omega_l$ is the laser angular frequency.

The phase shifts due to changes in the plasma density from eq. (3.3) are

$$\begin{aligned} &c_p \bar{n}_e L/\omega_l && \text{(first harmonic)} \\ &c_p \bar{n}_e L/(2\omega_l) && \text{(second harmonic)} \end{aligned} \tag{3.4}$$

where $\bar{n}_e$ is the line averaged electron density and L is the path length through the plasma.

As explained in section 3.1.1, after the beam crosses the plasma and receives its second frequency doubling, the original harmonic is filtered, leaving only the components and phase values $\varphi_1$ and $\varphi_2$ of both second harmonic components.

$$\begin{aligned} \varphi_1 &= 2(\omega_l t + \omega_l \Delta d/c + c_p \bar{n}_e L/\omega_l + \phi_1) \\ \varphi_2 &= 2\omega_l t + 2\omega_l \Delta d/c + c_p \bar{n}_e L/(2\omega_l) + \phi_2 \end{aligned} \tag{3.5}$$

These are the initial phases of the second harmonic, that when expressed in the intensities measured in the detector signal, the interference pattern can be modeled through

$$\begin{aligned} &I = I_1 + I_2 + 2\sqrt{I_1 I_2} \cos\left(\frac{3}{2}\frac{c_p \bar{n}_e L}{\omega_l} + \phi\right) \ \ with \\ &\phi = 2\phi_1 - \phi_2, \end{aligned} \tag{3.6}$$

which represents the model of the interferometer signal [51]. This model shows one disadvantage of some interferometers where, given that the sine is a monotonic function, it can only be estimated without ambiguity in its monotonic part and thus, reducing its usable range. It also shows clearly one of the main advantages of the dispersion interferometer's single path which allows for the canceling of the vibrational term $\omega \Delta d/c$ since both wave components experience the same vibrations.

It is important to clarify that even though $\phi$ is also a phase difference, it is just the constant offset level of phase difference that the signal will have regardless of the plasma. The real parameter of interest and relevant phase difference is the density change related factor.

To finish the model and remove the ambiguity of the sinusoidal functions, the modulation term coming from the photo-elastic modulator has to be considered. The phase is modulated to reduce the phase restriction due to the lack of monotonicity of the cosine function in eq. (3.6) and remove calibration necessities due to variations between discharges in $I_1$ and $I_2$. For this,

we express the intensity eq. (3.6) in terms of wavelength, add the modulation frequency $\omega_m$, substitute $c_p$ and reach a general expression of the line integrated electron density signal as

$$I(t) = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos\left(m\pi\sin(\omega_m t) + \phi + \frac{3}{2}\frac{\lambda_l e^2}{4\pi c^2 m_e \varepsilon_0}\int n_e \mathrm{d}L\right). \tag{3.7}$$

In this equation $m$ represents the modulation depth which is a gain selected on the photo-elastic modulator. It is important to consider that before the plasma discharge occurs, while the density is 0, an arbitrary phase difference value will be present and can be disregarded. The change in phase difference is what is measured making the measurement a relative value. This means for the analysis, that the offset of the phase $\phi$ before the measurement, assuming it is constant in a short time period, will not have an effect on the measured phase change. Also, $I_1$ and $I_2$ show negligible variations within a single modulation period if compared against the rate of change of the main parameter and therefore can be estimated for each period. With this we have our final desired model for the analysis of the interferometer measurement.

$$\begin{aligned} I(t) &= I_1 + I_2 + 2\sqrt{I_1 I_2}\cos\left(m\pi\sin(\omega_m t) + \frac{3}{2}\frac{\lambda_l e^2}{4\pi c^2 m_e \varepsilon_0}\int n_e \mathrm{d}L\right) \\ I(t) &= I_1 + I_2 + 2\sqrt{I_1 I_2}\cos(m\pi\sin(\omega_m t) + \Delta\varphi) \end{aligned} \tag{3.8}$$

This non-linear equation, with uncertainty on the cosine's ambiguity, is the final model required to describe the Dispersion interferometer. If we were to consider this model with typical values seen in the diagnostic and a sampling rate of 50 MSps which is the currently used, the resulting signal would look is shown in fig. 3.4.

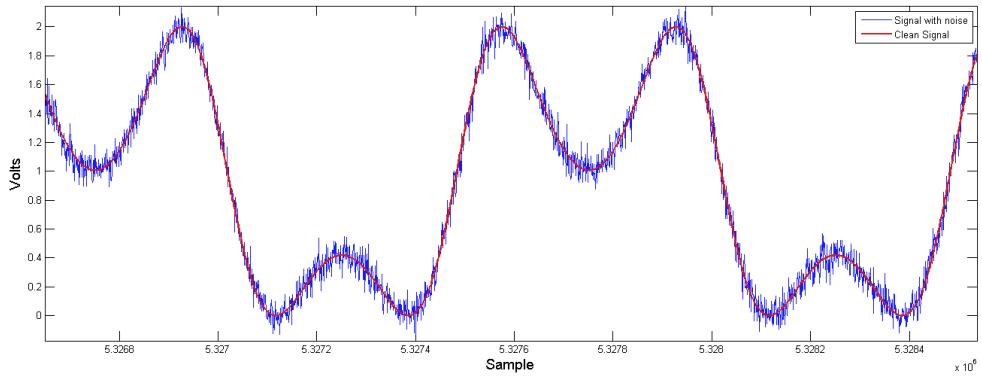



Figure 3.4.: Typical signal at the detector with 55 dB Gaussian noise, $m = 0.98$, $I_1, I_2 = 100\,\mathrm{mV}$, $\omega = 50\,\mathrm{kHz}$ and $\Delta\varphi = 1.5\,\mathrm{rad}$

In the following chapter we focus on defining the parameters of interest and selecting a Bayesian model and analysis approach to obtain the final line integrated electron density.

## 3.3. Bayesian Model for the Dispersion Interferometer

In order to gradually accelerate the computation of different time independent problems with a growing complexity, the DI was chosen as a proof of principle. Due to the non-linearity and ambiguity of the model, the DI meets the requirements where problem's complexity is still at the level of other plasma physics problems. The model and its parameters are known well enough to simplify the analysis to a single free parameter problem and start searching for an acceleration from a basic level. For this reason, the current section will focus on a 1-dimensional problem and use an approach of Bayesian analysis. This chosen approach is sequential Bayesian analysis as described in section 2.2.5.

From the previous section we know that the parameter of interest is the line integrated electron density. In eq. (3.8) is shown that this density is proportional to $\Delta\varphi$. Therefore, we will consider $\Delta\varphi$ as the main parameter of interest and the analysis will focus around it. We know that $\Delta\varphi$ is time dependent and it varies on each modulation period. Changes within them also occur due to smaller and faster density changes.

A target resolution of $1 \times 10^{17}\,\mathrm{m}^{-3}$ was initially selected. Due to the possible $\Delta\varphi$ resolution with current noise levels, phase changes within a single modulation period are not expected to surpass this uncertainty limit. This means that phase difference changes within a modulation period are not relevant, and we can neglect them.

With this approximation we can apply sequential Bayesian analysis to this signal within a period with $\Delta\varphi$ as free parameter.

The initial guess of the knowledge of the phase difference is defined as a flat prior distribution. Even though we know that the phase will start at the value representing zero density, the knowledge of the dynamic behavior for the density after that point is too poor to define a more constrained prior. Also, a zero density can be represented by an arbitrary initial phase shift within defined range as done in page 52. Therefore, a flat prior indicating lack of knowledge of the phase difference will prevent biasing the results and is expressed by

$$p(\Delta\varphi) = \begin{cases} 0 & \Delta\varphi < 0 \\ 1/2\pi & 0 \leq \Delta\varphi \leq 2\pi \\ 0 & \Delta\varphi > 2\pi. \end{cases} \tag{3.9}$$

Based on the parameter designs present in the previous section, the phase shift is expected to be anywhere between 0 and $2\pi$.

Since the truncation will naturally happen due to the selected prior, the likelihood is chosen

as a normal distribution of the data around the forward modeled value of the predicted voltage $V_{DI}$.

$$p(D_{\mathrm{DI}} \mid \Delta\varphi) = \frac{1}{\sigma_{\mathrm{DI}}\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}\left(\frac{D_{\mathrm{DI}} - V_{\mathrm{DI}}}{\sigma_{\mathrm{DI}}}\right)^2} \tag{3.10}$$

where $V_{DI}$ is the predicted value using eq. (3.8) as the forward model, and D are the measured data. As previously clarified, $\sigma_{\mathrm{DI}}$ usually represents the uncertainty of the measured signal.

Usually, variables that are not known with accuracy are ideally treated as free parameters and will be considered in more complex version of this analysis. Nevertheless, since the goal is to have a single free parameter for a proof of concept, they will be treated as known parameters.

All the uncertainties in these variables that are treated as non-free parameters will be considered. Foreseeing that as a first approach, calculations will be made in an FPGA without floating-point precision, every arithmetic errors in the forward modeling as well as the non-free parameter uncertainties are taken into account. This means that uncertainty contributions from the raw signal will be considered in $\sigma_D$, while the just mentioned error propagation is represented by $\sigma_V$. Thus, $\sigma_{DI}$ becomes

$$\sigma_{\mathrm{DI}}^2 = \sigma_D^2 + \sigma_V^2 \tag{3.11}$$

When dealing with a data set for which the evidence term $p(D_{DI})$ in eq. (2.1) is constant for all data points, we can disregard the term given that it does not require normalization for a search of the MAP. Therefore, using eq. (3.9) and eq. (3.10) the posterior distribution of eq. (2.1) can be changed to a proportionality:

$$p(\Delta\varphi_i \mid D_{\mathrm{DI}}) \propto \frac{1}{\sigma_{\mathrm{DI}}\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}\left(\frac{D_{\mathrm{DI}} - V_i}{\sigma_{\mathrm{DI}}}\right)^2} \frac{1}{2\pi} \tag{3.12}$$

where the subindex $i$ represents the $i-th$ sample. A common practice when doing this type of analysis is to calculate the logarithm of this function. It does not only reduce the computation time but also smooths the posterior distribution [52]. This simplifies eq. (3.12) to

$$\ln(p(\Delta\varphi_i \mid D_i)) = -\frac{1}{2}\left(\frac{D_i - V_i}{\sigma_{\mathrm{DI}}}\right)^2 + C \tag{3.13}$$

where its analytic solution leads to 2 possible answers for each time sample.

$$\Delta\varphi_e = \Delta\varphi_r \quad \text{and} \quad \varphi_e = \sin(\omega t) - \varphi_r + k2\pi \tag{3.14}$$

Here $\varphi_e$ is the estimated value of the phase difference, $\varphi_r$ the value due to the change in density and $sin(\omega t)$ the modulation frequency with $k \in \mathbb{Z}_+$.

As it will be shown in the following section, this scenario leads to a multimodality in the posterior that can be solved when good number of informative samples is available. For example our current scenario, where the parameter of interest can be considered constant within a single period.

$$p(\Delta\varphi_{1:N} \mid D_{1:N}) \propto \prod_{j=1}^{N} \left[ \frac{1}{\sigma_{\mathrm{DI}}\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{D_j - V_j}{\sigma_{\mathrm{DI}}}\right)^2} \right] \frac{1}{2\pi} \tag{3.15}$$

This can be approached through an analysis in bulk or a sequential analysis. Given that the samples are constantly entering and the bulk of data is not available, the sequential analysis takes place already introducing an acceleration idea of analyzing data while the next sample arrives.

## 3.4. Software Implementation

The software implementation of the analysis serves as a reference point to determine the acceleration possibilities. In this section the work described has no optimization in mind, that is, the code isn't optimized for processing speed. Due to the fact that this example was chosen simpler as a first step, not only the basic version in a Bayesian framework was developed, but a second version in C was done to benchmark against a code which optimizes processing time. The first is the implementation in Minerva to test the validity of the analysis and to carry out a profiling on where the code requires more time. The second version, after the analysis has been validated, is the C implementation which gives a better idea of how a faster version of this code can perform. This will be described in the following chapter in order to simplify the comparison for acceleration. Finally, in the last chapter of this first part of the project, the C version is compared against the designed FPGA accelerated solution.

### 3.4.1. Minerva implementation of the DI analysis

In order to observe how the likelihood and posterior function behave on the DI model, a version of it was implemented on Minerva using the tools provided by this framework while keeping the model unchanged. The first step was to analyze the posterior created by each sample in order to see the progression of the posterior when the sequential analysis approach is taken and each sample changes the posterior. A data set was generated where $\Delta\varphi = 1.5\,\text{rad}$ within one full period of its 50 kHz modulation frequency. With this data fig. 3.5 was generated where the posterior probability distribution is visible for each time sample in a single graph.

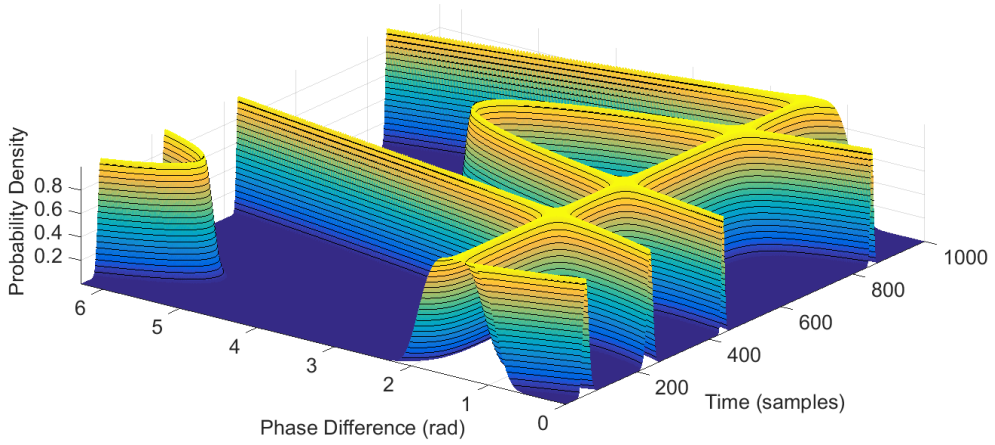



Figure 3.5.: Probability density for a full period. (1000 Samples for a simulated $\Delta\varphi = 1.5\,\text{rad}$.)

The search for an analytic solution to eq. (3.13) yields an oscillating solution as well as the

real one, representing the modulated phase plus the density term. This can be observed in fig. 3.5 where the posterior distribution has always two most likely values for each sample. The real answer in figure fig. 3.5 can be seen constant for all the time samples, while the oscillating solution can be seen changing along the period.

If we were to take a single time slice of the full period infig. 3.5, like the $440^{\text{th}}$ sample, to show the posterior from a single data point of the DI, we would see that each sample generates a multimodal posterior as shown in fig. 3.6.

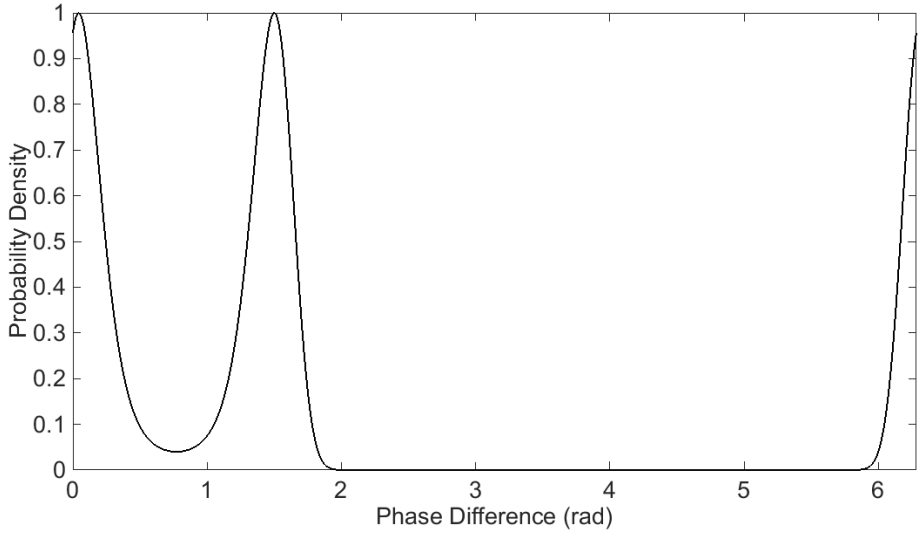



Figure 3.6.: Probability density for $\Delta\varphi$ at sample 440, showing the ambiguity when analyzing a single sample.

The fact that the posterior distribution is multimodal makes it nearly impossible to the finding of the maxima since they are equiprobable. This complicates the acceleration possibilities and in order to solve it, the sequential analysis of eq. (3.15) is carried out.

Within one period, from one sample to the next, the oscillating solution changes while the density phase change solution stays constant. This makes it possible that adding the information from all the samples will keep the likelihood of the density phase change high, and lower it for the oscillating solution, as shown in fig. 3.7.

At this point, given that the final distribution is of a low order, the maximum probability can be efficiently found with a simple scan of the posterior.

Regarding the uncertainty compared against expected values, the chosen phase resolution was $7 \times 10^{17}\,\mathrm{m}^{-2} \approx 0.03\,\mathrm{rad}$ which leaves the standard deviation of the final posterior well inside the desired range.

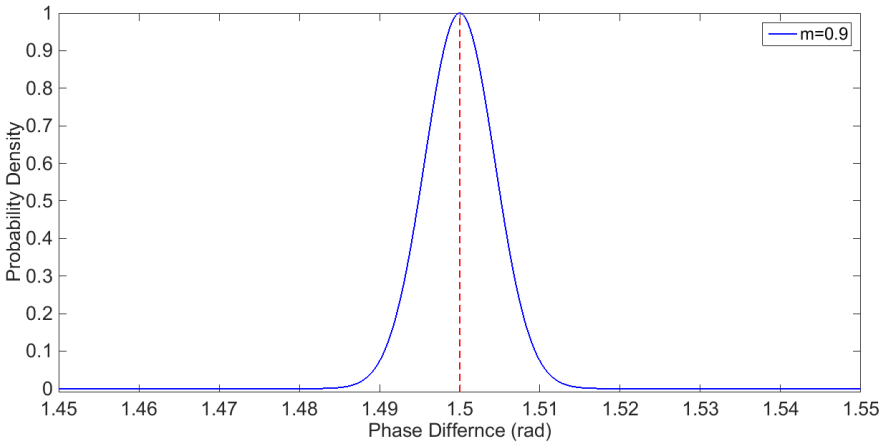



Figure 3.7.: Normalized PDF with all samples in a period and a 0.9 modulation depth.

Standard practices in Bayesian analysis require that each value that is not known with satisfying accuracy is handled as a free parameter. This makes it an extra dimension in the posterior of each parameter. However, to start tackling the possible acceleration, the dimensionality of the posterior was limited to a single dimension to learn how the analysis can be sped up in a simplified example of a complicated forward model. In case of this example, the parameter of interest $\Delta\varphi$ is treated as free, while the other parameters were included in the error propagation. This specific handling of other possible free parameters raises the question whether ignoring the other parameters, invalidates or affects the analysis.

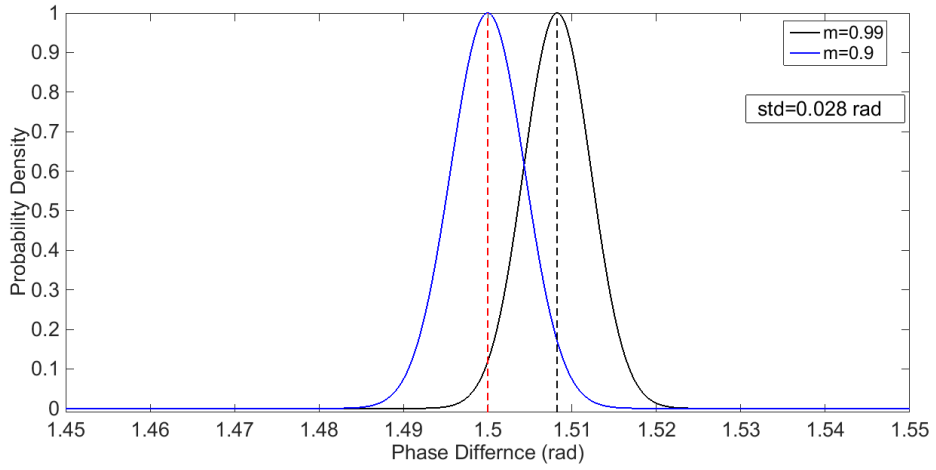



Figure 3.8.: Posterior with standard deviation for a 10% error in modulation depth $m$

As shown in fig. 3.4, the parameters $I1$ and $I2$ can be easily calculated within a single period. They also have little effect on the changes of the signal, making the modulation depth $m$ the main parameter considered known yet has indeed a higher uncertainty. Its value is entered manually in the photo-elastic modulator, however, its precision is not well specified by the

manufacturer.

In order to measure the effect of errors from the other non free parameters, $m$ was scanned since it is the one with the biggest estimated error. To test its effect on the global analysis, the posterior distribution was obtained with an introduced error on the value of $m$ which was selected to be 10% of its original value. By using synthetic data, the final phase value under these conditions could be compared to the synthetic value selected. The analysis in the Minerva model showed that the MAP has an error of 0.01 rad which is below the desired error and shows how for this simplified analysis it bears little effect to the final phase estimation. The next step is the hardware design after the software implementation of the data analysis was successful within the design parameters.

## 3.5. Hardware Design

Once the models have been selected and the software analysis has been carried out, the hardware acceleration is the next step. This model's analysis has the characteristic of requiring several calculations of the forward model that can easily be parallelized. Moreover, extensive mathematical operations that depend on different parameters can also be branched and parallelized to reduce processing time. The dependencies of the model on different incoming signals or parameters such as the modulation signal, gain from a processing platform that can respond immediately with parallel and independent processes. Due to these characteristics, an FPGA hardware implementation was chosen to make use of its high level parallelism and modular design capabilities.

There are three main ways to accelerate this analysis. These depend on the location of bottlenecks and the processing power required for each section of the software code. In the case of this general approach of Bayesian analysis, some acceleration can be achieved in the three main sections of the code. These are the evaluation of the forward model, the application of Bayes' theorem for N-dimensional posterior distributions and finally the inversion procedure. The following section describes the design of the analysis in FPGA architecture to achieve a speed-up with parallelization, modularization and pre-processing.

### 3.5.1. Acceleration of Bayesian analysis for the Dispersion Interferometer

When more than one sample is needed to produce a reliable result, the analysis method has to be chosen accordingly. In chapter 2.2 the method of sequential Bayesian analysis was introduced as means of analyzing incoming samples as they arrive, instead of a batch of preobtained samples. With this approach equation 3.15 was derived to apply sequential Bayesian analysis. This will be used to design an online circuit that updates the posterior with each incoming sample within a modulation period. The modularity and parallelization of the architecture allow the processing of received samples to occur simultaneously with the preparation for the next sample. This significantly reduces the processing time of the forward model.

The modularity and continuous processing are an advantage inherent in the dedicated hardware implementation of this analysis. The parallelism on the other hand, depends on the number of parallel processes achievable. That is, the resources available in the device versus the amount of resources required per parallel thread.

A Virtex 6 LX130T FPGA was chosen as the platform for this proof of principle. While

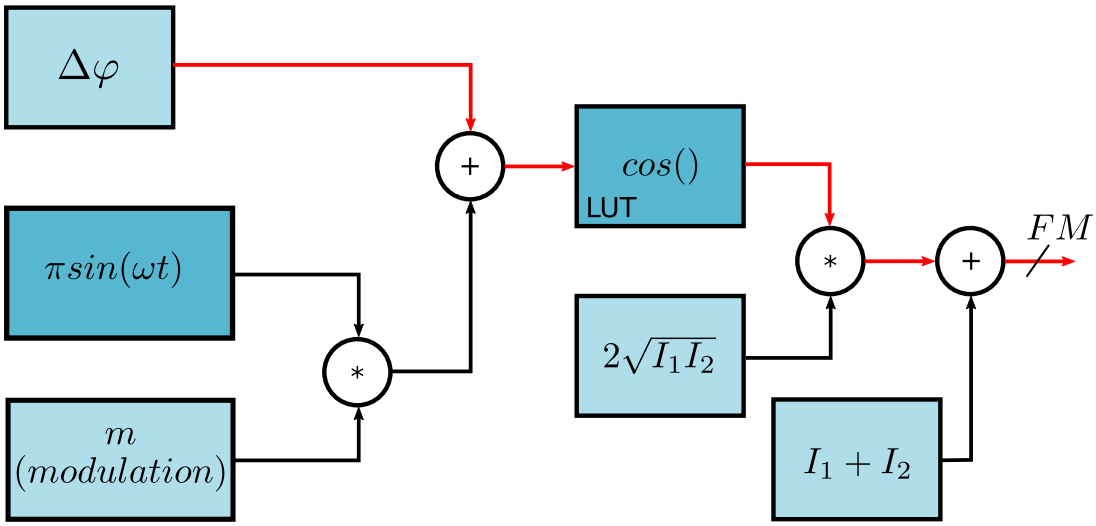



Figure 3.9.: Basic structure of the forward model (FM) and critical path (red) of the dispersion interferometer.

being one of the smallest chips in the family, this FPGA resources are enough to test a high level of parallelism. Since a scan along the phase difference range is required to create the distribution, the parallelism is applied in the evaluations of the forward model for different values of $\Delta\varphi$. It is exploited by creating a thread for each of the values of the $\Delta\varphi$ range, as opposed to the software approach where each value of the scan is done within a loop. The parallel calculations are done with a resolution that gets close enough to the desired value of $\approx 0.03\,\text{rad}$ while staying within the FPGA's resource capabilities. The example selected tests how much can be gained through a high parallelization in a relatively simple scenario of low dimensionality like the DI.

#### Resolution and error propagation design

The selected and tested forward model was the first section to be parallelized by designing threads that calculate each iteration of the forward function, eq. (3.8). The basic operations for the forward model are shown in fig. 3.9. Most of these parameters are available before the sample arrives. The modulation factor is set manually in the modulator and the sine function phase and frequency are known through the modulator trigger signal. Finally, the $I_1$ and $I_2$ parameters are combined as offset and amplitude of the function. As discussed in the previous chapter, these change negligibly within time scale of two or three periods. A measurement of amplitude and offset of the previous period can be used as values for the forward model while considering the introduced error.

This means that these threads of the forward model can also be timed and modularized so

that their output is precalculated and ready before the next sample arrives. The number of parallel calculations depends directly on the resolution of the $\Delta\varphi$ scan desired. Nevertheless, to determine limits of resource usage, the implementation is done with an initial value of 60 parallel FM's. This equates to resolution similar to the design constraint of final phase error. This target resolution is of $7 \times 10^{17}\,\mathrm{m}^{-2} \approx 0.03\,\mathrm{rad}$.

To determine the resolution and error of the hardware implementation, two main design factors have to be considered. The first one is the already mentioned resolution of the scan of $\Delta\varphi$. The more values calculated in the scan, the higher the resolution. The second one is the arithmetic precision given the word length of each parameter, which is tied to the parameters known error. The word length design is therefore chosen so that the error introduced by the arithmetic precision is smaller than error due to the scan resolution of $\Delta\varphi$. The scan resolution and the uncertainty of the Bayesian analysis is thus taken as the final resolution.

This approach varies from the software approach in that this error due to the precision of the analysis implementation is introduced. All calculations with a given word length have an upper bound relative error that comes from a bit truncation in arithmetic operations. In computers this is typically known as the machine epsilon. With double-precision floating-point arithmetics, this error tends to be low enough to consider the results in the forward model to be exact.

For the case of the FPGA implementation, double-precision floating-point arithmetic operations can be implemented. Nevertheless, it is not clear its effect on the FPGA resource consumption given the model's size. A part of this project's aim is to define whether this and bigger models are feasible as a FPGA implementation.

Given that the final resource usage is hard to estimate at this point, a lower resource consuming architecture was preferred. Fixed-point arithmetic architecture with minimum word length for each parameter suits this purpose adequately.

This low resource hardware implementation with limited arithmetic precision requires accounting for the error introduced in the forward model. With the knowledge of each parameter's error, its word length can be designed to match accordingly. This analysis of error propagation in the forward model is done to ensure that the final arithmetic precision is better or matches the one limited by scan resolution. By doing this, the parameter word lengths can be selected to satisfy the final resolution design constraint.

Given that several samples are used in one period to calculate the final value, and that this value must satisfy the final desired error, the error propagation is done backwards to define the required word lengths. With it, the arithmetic precision for each parameter is ensured to be the minimum possible in order to reduce resource usage per thread.

With the use of eq. (3.11), we can calculate the word length required for the buses. This starts by relating numerically the standard deviation of the posterior after analyzing all the samples, to a single likelihood function. With the target posterior resolution, the value of uncertainty required in the likelihood is 0.032 rad. Going backwards in the error propagation of the forward function $V_{DI}$, we have

$$\sigma^2_{V_{DI}} = \left[\frac{\partial V}{\partial I_1}\sigma_{I_1}\right]^2 + \left[\frac{\partial V}{\partial I_2}\sigma_{I_2}\right]^2 + \left[\frac{\partial V}{\partial cos}\sigma_{cos}\right]^2 \tag{3.16}$$

If we approximate the error in the argument of the cosine to the error in the cosine through a Taylor expansion, we can show that $\sigma_{cos} \approx \sigma_A$, with $A$ representing the argument of the cosine, and therefore

$$\sigma^2_A = \left[\frac{\partial A}{\partial m}\sigma_m\right]^2 + \left[\frac{\partial A}{\partial \omega}\sigma_\omega\right]^2 + \left[\frac{\partial A}{\partial t}\sigma_t\right]^2 + \left[\frac{\partial A}{\partial \Delta\phi}\sigma_{\Delta\phi}\right]^2 \tag{3.17}$$

Using this equation for the error propagation we can introduce the resolution of the free parameter $\sigma_{\Delta\varphi} \approx 0.31$ and the estimated errors in each of the non-free parameters to define the required word lengths. This includes the jitter of the trigger signal from the modulation as well.

With this analysis we can make sure the errors on the non-free parameters and the arithmetic precision do not affect the target uncertainty for the phase difference.

This can be summarized in three values. The final value of eq. (3.16) will define the maximum precision achieved in the calculation. The resolution will be limited by the number of threads in the scan of $\Delta\varphi$. Finally, the uncertainty will be defined by the standard deviation of the posterior distribution obtained with the analysis. This constrains the final uncertainty which cannot be better than the previous two limiting values of resolution and precision.

### Design of Hardware Architecture

Once the error propagation is done, the word lengths can be defined. The forward modeling of the cosine's argument is composed of several stages based on eq. (3.18).

$$I(t) = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos(m\pi\sin(\omega_m t) + \Delta\varphi) \tag{3.18}$$

Starting with $\Delta\varphi$ scan, these parameters are implemented as an array of registers. The array depth defines the resolution of the parameter scan and the number of parallel forward models to be implemented. This approach allows us to have $N$ copies of the forward model depending on the desired resolution and available resources. The selected $N = 60$ parallel forward functions

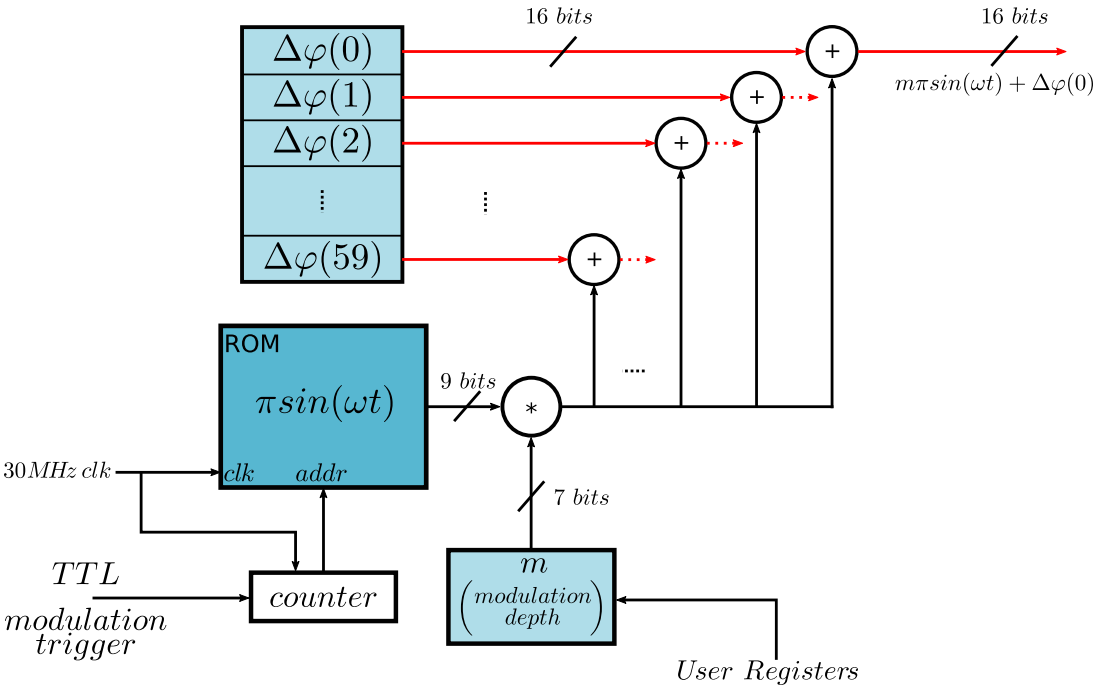



Figure 3.10.: Hardware design for the cosine's argument calculation. Based on the error propagation the minimum word lengths in the forward model are defined.

for each value of $\Delta\varphi$ keeps the resource usage low while staying within the desired resolution. Each forward function has a final word length of 16 bits and matches that of the incoming samples.

Parallel to the $\Delta\varphi$ registers, the modulation factor is calculated as seen in fig. 3.10. The sinusoid for the modulation factor is generated by a Read-Only Memory (ROM) that holds the scaled values of $\pi \sin(\omega t)$. Given the periodicity of this factor, the sine can be synchronized with a **Transistor-Transistor Logic** (TTL) trigger signal coming from the modulator described in 3.1.1. With this trigger, a counter is synchronized and used as an address for the memory that outputs the respective sine signal.

The sinusoidal output of the ROM is multiplied by a register that holds the scaled value of the modulation depth $m$. This value can be modified through user controlled registers in order to match it to the manually selected setting on the modulator. These registers are modified via a **peripheral component interconnect express** (PCIe) bus present on the FGPA and connected to a local computer.

The last operation needed to finish the cosine's argument is the addition of the phase value. With the full argument ready, a method for the implementation of the cosine is required.

A fast and generic approach to function estimation is an LUT that contains the range of desired evaluations of the function. Given that this project aims to achieve acceleration for any

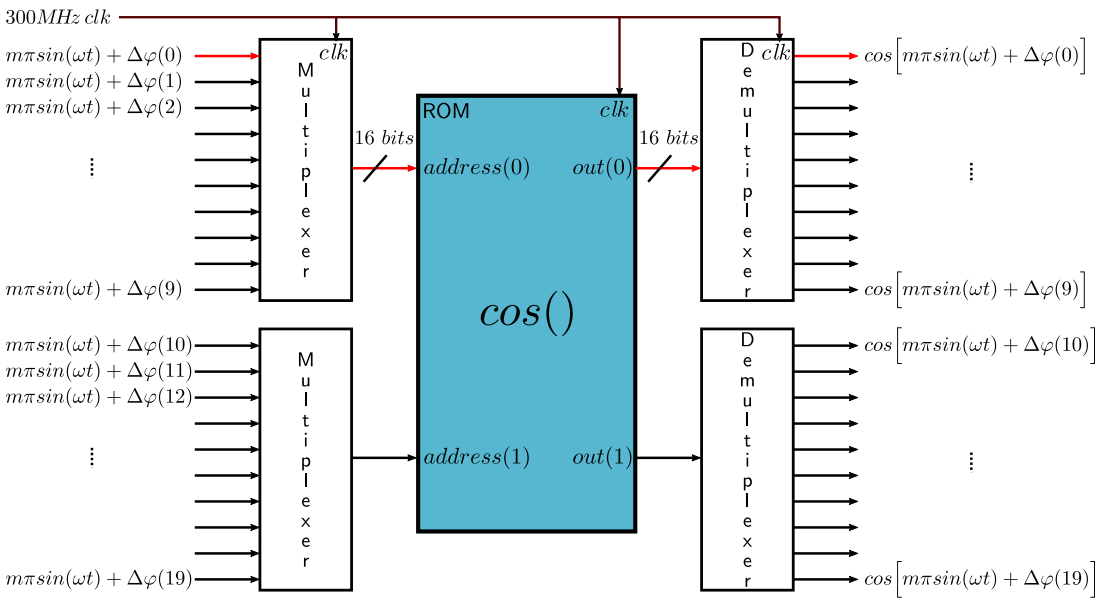



Figure 3.11.: Multiplexing scheme for the sharing of the cosine's LUT. The multiplexed cosine estimation section is driven at 300 MHz to match the 30 MHz data-path clock.

given model, the use of an LUT falls withing the generic solution picture and was therefore initially preferred.

This requires that the values of the argument are scaled adequately to satisfy the addressing of the LUT memory space. To meet the required resolution a 16 bit address is used and scaled with

$$K(m\pi \sin(\omega t) + \Delta\varphi) + B \tag{3.19}$$

where $K$ and $B$ are scaling and offset constants that depend on the desired resolution or memory depth. These are $K = 5452$ and $B = 20553$ for a memory address space between 0 and $2^{16} - 1$.

Considering the scaling and the error propagation analysis, the word lengths of each parameter are: 7 bits for $m$, 9 bits for $\pi sin(\omega t)$ and 16 bits for $\Delta\phi$. The final word length would then match the 16 address *bits* for the cosine evaluation of a given argument.

For complex functions, the LUT calculation of 60 parallel threads would be infeasible given that the resources would not suffice. Therefore, the cosine estimation LUT is shared by a number of multiplexed threads. The workload is divided between 3 ROMs that require 30 blocks of $36k$ RAM resources each. To reduce the consumption of resources a Dual ROM is used. This ROM has two address and output ports that allow the reading of two memory locations simultaneously. Due to the dual access, 10 values of the cosine's argument are multiplexed to

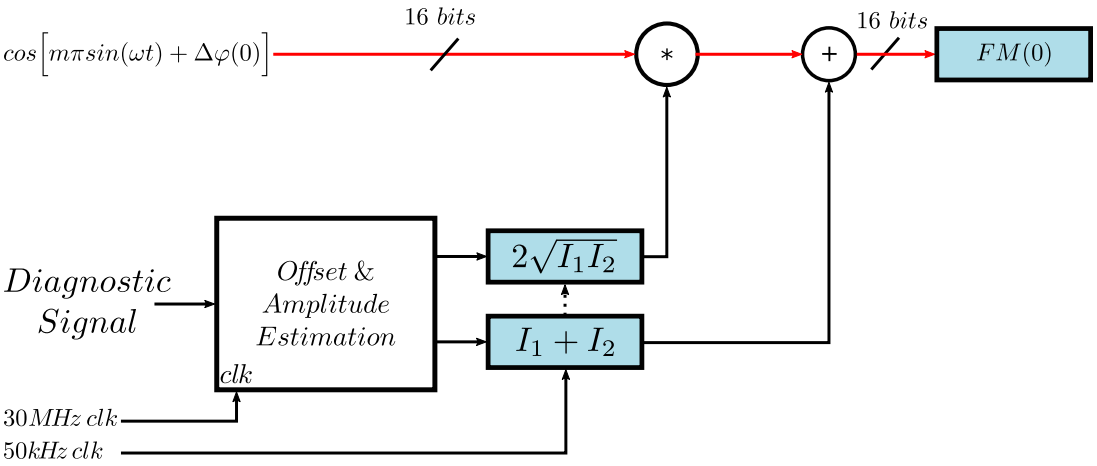



Figure 3.12.: Final section of the forward model. Based on the measurements of the previous period, the offset and amplitude are estimated and applied. A 15 bit rescaling is carried out after the amplitude is multiplied.

each ROM input as fig. 3.11 shows.

This multiplexing introduces a limitation and sets the main data-path clock frequency of the design. Given that the Dual ROM has a max frequency of 400 MHz, a test 300 MHz multiplexing and read clock is selected to simplify FPGA timing in the initial design. Since the argument values are multiplexed with 10 values per ROM address port, a 30 MHz main data-path clock is used to comply with this design limitation.

Finally, the $I_1$ and $I_2$ values are included by using the fact that these do not vary significantly from one period to another. The selected approach is to constantly measure the input signal and estimate a maximum and minimum for each period, as seen in fig. 3.12. With it the offset $I_1 + I_2$ and amplitude $2\sqrt{I_1 I_2}$ are determined and the possible introduction of an error is considered in the error propagation. The values estimated from the previous period are then applied to the following period's forward model.

As mentioned in the previous section, a significant advantage of this design is the modularity of different processes on the FPGA running in parallel. With a parallel module dedicated to the forward model, the cosine factor and its arguments are calculated independently and constantly. The complete forward model can then be pre-processed before the next sample arrives.

Once the forward model is done, the calculation of the likelihood function can take place with the incoming sample.

$$ln(p(\Delta\varphi_{1:n} \mid D_{1:n})) = -\frac{1}{2\sigma_{\mathrm{DI}}^2} \sum_{i=1}^{n} \left[ (D_i - V_i)^2 \right] \tag{3.20}$$

As discussed in section 3.3, the use of the logarithmic form of Bayes' theorem changes the

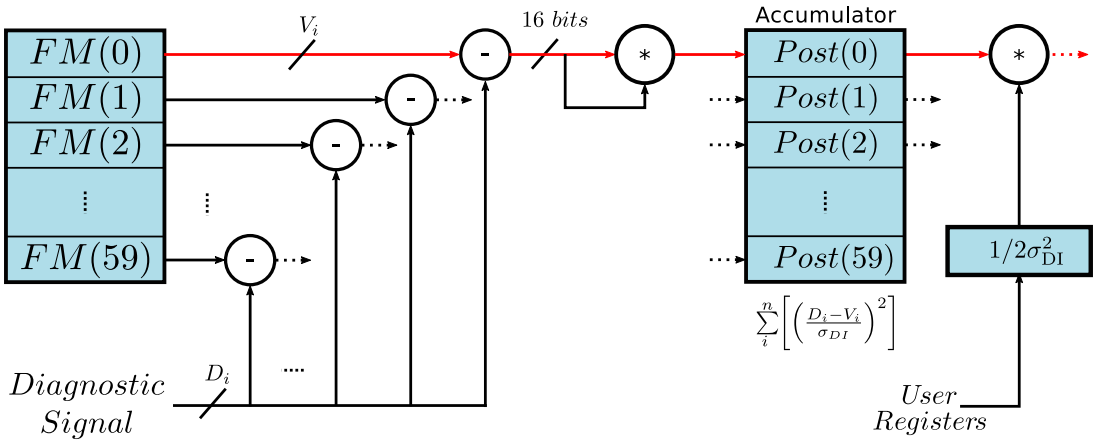



Figure 3.13.: Design of the application of logarithmic version of Bayes' formula and final sum. After this section, the final step is the inversion to define a most likely value.

function to eq. (3.20) while still having the same MAP. This logarithmic version is the same formulation used as standard in Minerva.

This calculation of the likelihood, when compared to eq. (3.12), is much simpler and reduces the required number of operations. An incoming sample is compared against each forward model result by their subtraction given the chosen likelihood model. It is followed by a final multiplication calculating the squared value required to generate part of the likelihood distribution for a single sample.

The calculation of the posterior starts with the sum of each likelihood. The stored data will continuously accumulate the results of a likelihood function for each incoming sample until a full period is analyzed as seen in fig. 3.13.

After the 200 samples in a period have been accumulated, a division over the factor containing the global uncertainty ($\sigma_{\mathrm{DI}}$) is necessary. This uncertainty value is obtained by considering the error propagation analysis and signal noise. Its inverse can be pre-calculated and set through user controlled registers in order to substitute and multiply with $1/2\sigma^2_{\mathrm{DI}}$. By taking this factor out of the sum, it will be applied only once every 200 samples and thus reduces resource consumption and processing time.

After the array that accumulates the likelihood values, the prior defined in eq. (3.9) is added as an initial probability. Since the prior is only taken into account once per data set (a full modulation period), it is not required to keep applying it for every incoming sample. As previously defined for this case, the prior value is a flat distribution that takes the same value for every scanned value of $\Delta\varphi$ and was modeled as $1/2\pi$.

The full posterior is ready and the inversion, or search for the most likely value, takes place. This is estimated by comparing each value with its array neighbor, meaning that the 60 values

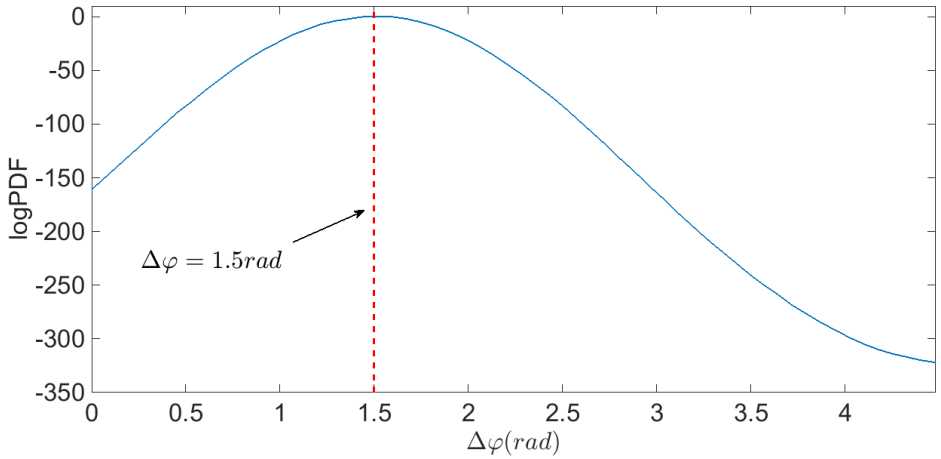



Figure 3.14.: Unnormalized posterior distribution of the hardware implementation with simulated $\Delta\varphi = 1.5\,\text{rad}$.

require 6 comparator levels until the maximum is found.

It is often the case that besides the MAP, the full posterior distribution should be observed and analyzed. For this purpose, the final design's real-time results are monitored using Xilinx's Integrated Logic Analyzer (ILA) tool. With it, the final distribution and its temporal behavior can be observed.

### 3.5.2. Results and Analysis

#### Verification of validity and accuracy of the results

In order to verify the validity and accuracy of the implemented design, two test bench scenarios were generated. The first test signal is a synthetic version of the detector signal with the same conditions used in software analysis in Minerva. It has 55 dB of Gaussian noise which introduces variations in the offset and amplitude, modulation depth $m = 0.9$ and $\Delta\varphi$ =1.5 rad. In this test, the MAP of the posterior obtained in the hardware implementation is required to match the phase difference used to generate the signal.

Figure 3.14 shows the posterior distribution with the shape of a truncated Gaussian covering the phase difference range selected for the 60 values of $\Delta\varphi$. The truncated section is the rest of the $2\pi$ periodicity of the signal that falls outside the required range. This initial verification shows how the MAP matches the synthetic value selected. As tested for the software approach, it also removes the effect of multimodality present in fig. 3.6, resulting in a posterior with a single maximum.

The second test bench verifies proper estimation within the required $\Delta\varphi$ range of 0 to

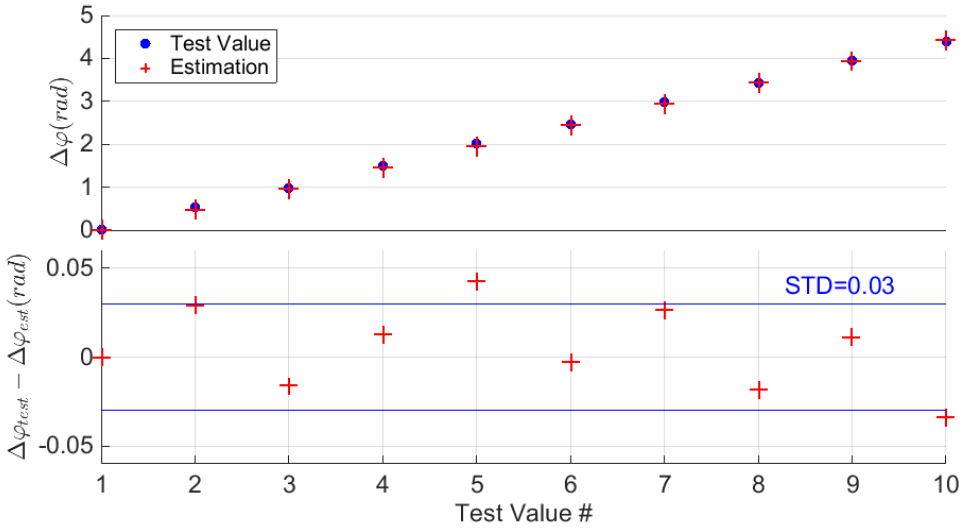



Figure 3.15.: MAP and STD measurements for $\Delta\varphi$. Test and hardware estimated values (top). STD values with respect to design requirements (bottom).

4.4406 rad with a resolution of 60 values. It is also aimed to test whether the respective uncertainties of the maxima within this range, meet the design requirements. For this, a test signal was created with the same noise conditions and a changing phase difference. Here $\Delta\varphi$ is a time dependent function with 10 different values along the required range.

Figure 3.15 shows the results of this benchmark where the original phase value and the estimated one are compared to see if the desired uncertainty was met.

From these 10 test values, only 2 fell outside the target standard deviation desired with an acceptable error margin. The other 8 fall within the required uncertainty, validating the estimation of the maxima as well as the error propagation analysis in the hardware implementation.

### Acceleration comparison against software approach

Once the hardware implementation results have been verified, we can analyze the achieved acceleration when comparing against the typical software approach.

Before a global speed up review, it is important to discuss the optimization level of the hardware implementation. This first stage of the project serves as a proof of principle that probes if it is feasible to tackle more realistic and complicated problems. In this sense, the first stage shows the expected validity which was reviewed in the previous section. An acceleration was also achieved and described in the following analysis. Nevertheless, the current FPGA

design shows room for further improvement in terms of clock speed optimization which will be also discussed.

In order to measure acceleration, a specific implementation must be chosen as a comparison point. Comparing against current accelerations of the many forms of Bayesian analysis can present some difficulties given the diversity of solutions that target specific analysis methods. As previously mentioned, this work's analysis differs from other solutions such as Bayesian filtering and smoothing filters. In this work the analysis is applied to the general use of Bayes' theorem and forward modeling. It is specifically targeted at, but not constrained to, complex physics problems where the dynamics of the phenomena are not well know.

Developed implementations of FPGA accelerators are, for example, applied to Bayesian networks where the hardware is mainly used to evaluate and compare score values of different models [53]. This means that the comparison is better done against the analysis carried out in Minerva. Given that the size of the model and its respective code are considerably small, a basic implementation of the algorithm in Minerva (Java) was translated to C and profiled to benchmark against the FPGA implementation.

The comparison of the design's processing time introduces the complication that the hardware has an independent module that constantly pre-calculates the forward model. This essentially means that the forward model is done before the sample arrives to the FPGA. It does not play a role in the total processing time of each sample in a full period and thus in the duration of the total analysis. Due to this, both processing times with and without forward model duration are considered.

Starting with the processing time for one sample, fig. 3.16 shows how the forward model itself requires 8 clock cycles to generate a value for each of the 60 parallel threads scanning $\Delta\varphi$. The application of Bayes' formula (eq. (3.20)) is achieved within 4 clock cycles per sample if the final operation of the uncertainty is considered. While the uncertainty factor is applied after the 200 samples have been considered, it is required if a posterior would be created from one sample. The last part is the estimation of the MAP from the 60 values that make up the final posterior distribution, which requires 6 clock cycles. At a 300 MHz clock, counting from the sample arrival until Bayes' formula plus MAP are completed, a total of 10 clock cycles or 0.33 µs have elapsed. If the pre-processing of the forward model is ignored and its processing time considered, a total of 18 clock cycles or 0.6 µs are needed.

In terms of the full analysis, a set of 200 samples (full modulation period) is required to provide a phase difference value. This period takes 20 µs and the pipelining of the architecture allows it to process each sample as it is received. This means that a full analysis lasts 20.33 µs

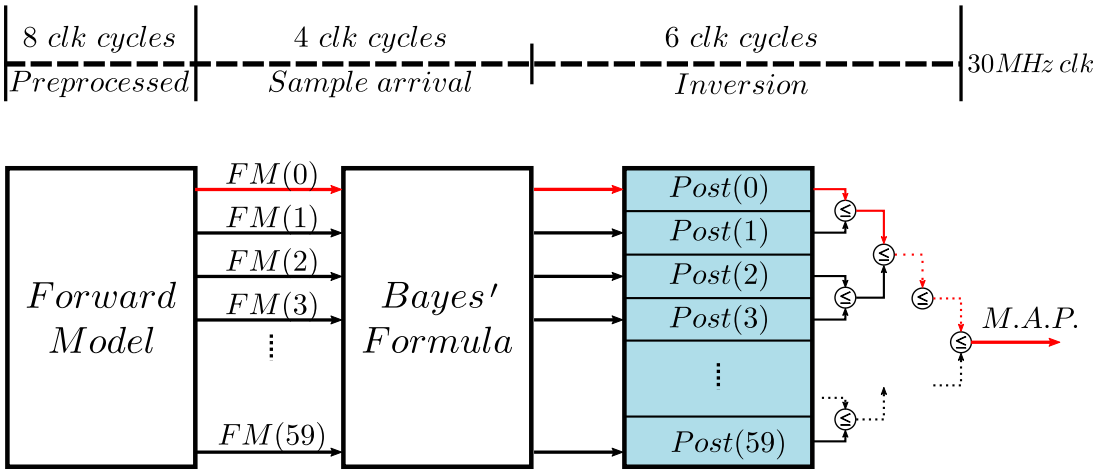



Figure 3.16.: Global scheme of each analysis section and their respective duration for the generation of a posterior from a single sample.

when the time for all the samples to arrive and the time required to analyze the final sample are considered. Thus, any reduction in the modulation frequency could reduce the 20 µs period and yield a faster processed data rate.

Regarding the software implementation, the Minerva framework is not optimized for real time processing. The C code was therefore developed to calculate the same analysis as the Minerva code without having to deal with the inner processes of the framework. This means it follows the exact same core algorithm used in the Java code in Minerva and thus only composed of the same basic operations used by the FPGA code.

As in Minerva, C code uses the double data type which has a floating-point double-precision that does not need the error propagation analysis done in the hardware implementation. It is executed on an Intel $i3-4130$ CPU with a 3.40 GHz clock speed. The analysis for a single sample is then executed 50000 times to determine an average per execution and compare against the time required for a single sample in the hardware implementation. Under these conditions, the C code yielded an average of 5 µs per sample on a single thread. If compared against the FPGA implementation with a pre-processed forward model, the FPGA achieves a 16-fold acceleration. In the case where the forward model is included, an 8-fold acceleration is reached.

While this acceleration is already promising, it is important to analyze it in the frame of possible optimizations in both implementations. In the FPGA approach, this first implementation shows that the main bottleneck is the cosine function evaluation in the forward model. It limits the data-path clock given its considerable resource consumption and maximum read rate. The resource consumption causes the design to share the LUTs by multiplexing the parallel threads use it. This consumption is visualized when analyzing the resource usage shown in table 3.1.

Table 3.1.: Use of FPGA resources.

| Resource | Utilization | Available | Utilization(%) |
|---|---|---|---|
| Registers | 4503 | 160000 | 2 |
| RAMB36-18 | 211 | 528 | 40 |
| DSP48 | 91 | 480 | 18 |
| Slices | 2066 | 20000 | 10 |

An increase in the number of cosine LUTs in the forward model would reduce the multiplexing degree, allowing the use of a higher frequency clock in the main data-path.

Regarding read rate, the cosine function estimation was read at 300 MHz. The Block Memory Generator IP-Core used to generate the LUT defines a max clock frequency of 450 MHz. By clocking the LUT at a higher frequency, the data-path clock can also be improved. This means that a considerable increase of the main clock is possible by combining the resource increment, a lower level of multiplexing and use of the maximal read rate. This optimization review is focused on showing that the FPGA has more room for optimization than the C counterpart. Its parallelization level, which is the main acceleration factor, is only limited by resources whilst the C code tends to have a lower number of threads limit.

An alternative solution to increase its clock speed is with the use of a less consuming cosine estimation method. The LUT approach was preferred over **coordinate rotation digital computer** (CORDIC) or other algorithms that can be clocked at higher frequencies. The reason for doing this was to keep the design general enough for other possible models that use functions that cannot be implemented through CORDIC [54, 55]. Complex functions will present, in general, difficulties with regard to resource usage or clock cycles required along the critical path. Therefore, a LUT table presents a viable solution that can be generally applied to several functions in order to assess timing and implementation feasibility. Another LUT advantage is that any function evaluation requires a single clock cycle read. This comes at a cost of a greater resource consumption when higher function resolution and precision are needed.

#### Other architecture considerations

Regarding implementation possibilities, this conservative first FPGA approach clearly is at a disadvantage when considering the limited fixed-point arithmetic precision. While it introduces a reduction in resource usage, the uncertainty requires complex error propagation analysis to satisfy a specific uncertainty. For bigger and more complex models, this cumbersome requirement could present difficulties.

The advantage of a software implementation with double precision floating-point operations motivates to match its precision while still maintaining an acceleration for the second part of this project. This way, the final uncertainty depends mainly on the used priors and the incoming data. An increase in arithmetic precision represents a usual trade off in terms of precision versus parallelism when working with limited FPGA resources. This will be analyzed in the frame of more complex problems in the following chapter.

Finally, for cases of a low dimensionality posterior, the parallel approach and the simple solution of a comparator scheme for the search of the MAP is possible. Nevertheless, this reduced interferometry model shows that this is rarely the case. In this already small model, free parameters had to be ignored to intentionally keep the dimensionality low. Multidimensional posteriors require alternative solutions.

For a low number of dimensions, direct search algorithms like Hooke and Jeeves are a straight forward alternative to find the MAP in different non-linear optimization problems as a first approach [35]. For higher dimensionalities or more complex distributions, sampler algorithms like MCMC provide significant samples from which a MAP can be found by generating a histogram. Given that iterative approaches inherently increase the processing time, accelerated solutions for this also exist like Parallel Tempering MCMC described in section 2.3.

# 4. Accelerating Data Analysis for Temperature and Density Profiles

## 4.1. W7-X Thomson Scattering System

For the sake of a better visualization and understanding of the model behind, the W7-X Thomson scattering system is briefly introduced. The physics model is described giving enough information about the diagnostic design and operation, so that the selection of the forward model and the physics principles behind it are understood.

### 4.1.1. Diagnostic Construction and Operation

The Thomson scattering diagnostic measures laser light scattered by free electrons in the plasma. This scattered light provides information about electron density and velocity distribution, i.e., temperature.

It is an active diagnostic that causes negligible perturbations when obtaining a measurement given that it launches a high energy beam into the plasma, a tiny fraction of which is scattered by the free electrons.

If we were to describe the W7-X Thomson scattering measurement as a process, it would start with a light beam from a periodically pulsed Nd:YAG laser [56]. The laser has a fundamental

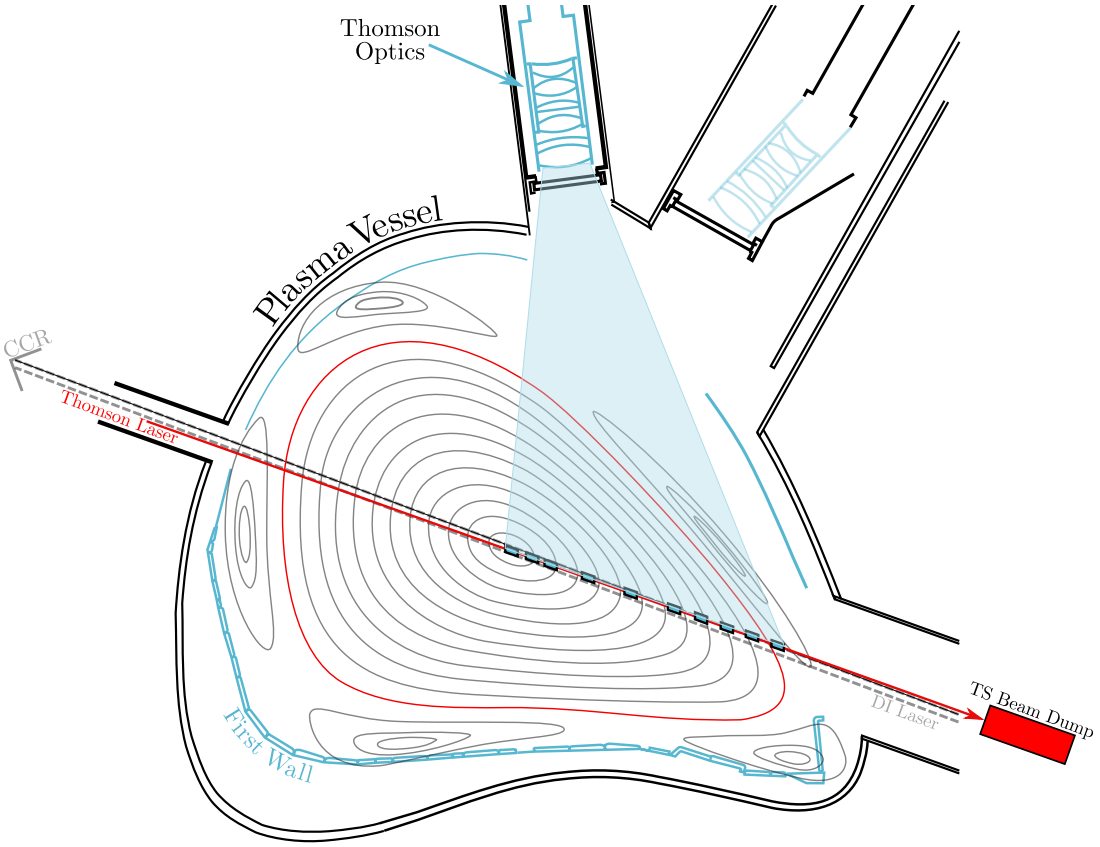



Figure 4.1.: Thomson scattering diagnostic cross-section showing laser path (red), scattering volumes (aligned blue squares) and plasma magnetic surfaces (concentric paths).

wavelength of $\lambda = 1064.14\,\text{nm}$ with a tunable energy from 0.7 to 2.5 J, with a repetition rate of 10 Hz. The laser beam is then guided from a separate room to the torus hall, the hall where the W7-X is placed. Using a set of mirrors, the laser beam is guided to the entrance port of the plasma vessel.

The laser beam passes through the center of the plasma where part of the light will be scattered the free electrons of the plasma. The rest of the laser energy is collected by a beam dump behind the output port located and aligned opposite to the entrance port as shown in fig. 4.1.

The scattered light is collected by a lens module that observes the entire plasma cross-section of 1.6 m along the beam line. The optics are located in air inside an immersion tube that isolates them from the vacuum vessel. A support structure, which is decoupled from the stellarator for mechanical stability, holds the lens module in place. The optic set used in this work and in the first operation campaign, observes the outer half of the plasma cross-section with 10 scattering volumes (spatial channels) along the laser path from which scattered light is collected. The arrangement of these volumes can be seen in fig. 4.1.

The scattered light from these volumes is collected by dedicated rectangular fiber bundles, which define the relevant dimensions of the scattering volume. These fibers transport the collected light back to a separate room for analysis.

The mentioned room contains several polychromators, as depicted in fig. 4.2, to which each fiber bundle connects.

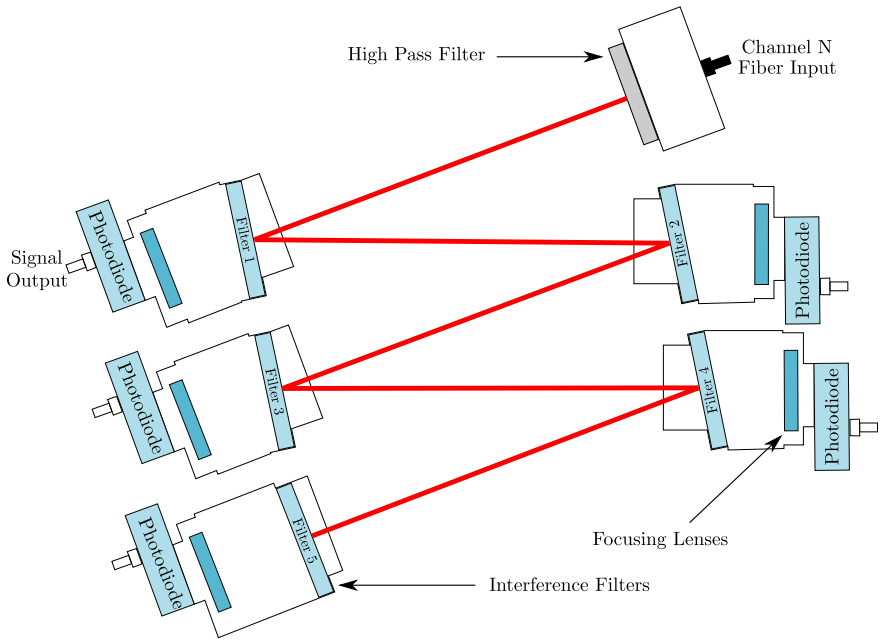



Figure 4.2.: Spectral channels and filters of the polychromator for one of the Thomson spatial channels.

The main function of these polychromators is to separate the scattered light into five spectral segments, while maintaining an acceptable signal-to-noise ratio. This in principle serves the same purpose as a spectrometer with a limited spectral resolution. The spectral coverage of the polychromator channels is exemplified by fig. 4.3.

In a polychromator, the light from the fiber bundle passes through a set of lenses and interference filters that separate the light into the spectral segments. In each channel the light is detected by a silicon avalanche photo-diode.

Finally, a set of ADCs digitizes the signals from all channels and stores them for further processing. Given the scattered light spectrum and 10 ns duration of the total pulse, the ADCs have a bandwidth of 300 MHz with a dynamic range of 14 bits and 1 Gsps temporal resolution. The process of taking a measurement with the Thomson scattering diagnostic is completed with the collection of these samples. With the description of the diagnostic finished, the review of associated physics, modeling and data analysis can be carried out.

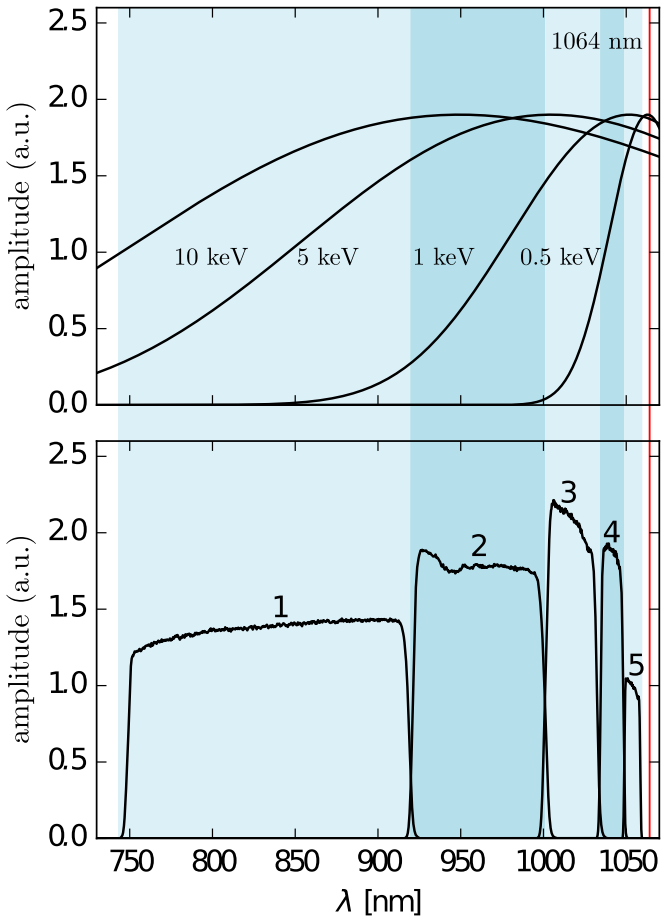



Figure 4.3.: Top: Spectral distribution at different temperatures. Bottom: Spectral coverage of the 5 polychromator filters. The shades of blue show the coinciding sections. The laser wavelength (red) at 1064 nm is also displayed.

## 4.2. Thomson Scattering Principle and Modeling

The required conditions for electromagnetic emission by free charged particles, which is the case of interest, is that said particles change their velocities due to the influence of a magnetic or electric field. In plasmas, an example of this is an electron moving due to the Coulomb field of an ion and generate what is named bremsstrahlung through acceleration.

The Thomson scattering diagnostic measures the dipole radiation from electrons accelerated by intense lasers fields. These electrons are driven to oscillate at the same frequency as the probing electromagnetic wave. In this case, the mentioned Nd:YAG laser is the electromagnetic wave responsible for accelerating the electrons while the induced oscillation causes it to scatter a photon.

When modeling Thomson scattering, the entire model can be divided into two parts, the physics model of the scattering process occurring in the plasma and the diagnostic component-specific parameters. The latter considers all the required changes in the measured signal introduced by the components of the diagnostic after the scattering process. It includes lenses, fibers, transmission efficiencies, electronic gains and the spectral filtering done by the polychromator filters.

Starting with the physics model, the scattering can be described as a sum of unrelated single electron-photon scattering processes. This means, that by using a model describing the scattering process of an electron under the influence of a polarized wave's electromagnetic field, we can add the contribution of each electron within a volume of plasma and calculate the total irradiated power from a scattering volume.

Considering the case of a single electron, the thermal motion of the electron will directly affect the frequency of the scattered wave due to the Doppler effect: the moving electron experiences the incident wave at a different frequency and Doppler shift compared to the observer point. The velocity distribution of the electrons will thus determine the Thomson spectrum. The density on the other hand, is proportional to the amount of photons collected by the lens for a specific solid angle [57]. These considerations of density and velocity distribution allow us to model the intensity and spread of the Thomson spectrum.

It is also important to take into account the statistical distribution of electrons in the plasma. The W7-X electron temperatures are high enough for the model to require a relativistic correction of the equations ($T_e > 1\,\mathrm{keV}$).

With all the aforementioned factors considered, we can describe the differential scattered

power in a general form as:

$$\frac{\mathrm{d}^3 P_s}{\mathrm{d}\Omega_s \mathrm{d}\epsilon \mathrm{d}V} = r_e^2 n_e \langle S_l \rangle S(\epsilon, \theta, \alpha), \tag{4.1}$$

where on the left-hand side,$d^3P_s$ is the differential scattered power, $d\Omega_s$ is the solid angle differential, $d\epsilon$ is the wavelength shift differential, $dV$ is the volumetric differential. On the right-hand side $r_e$ is the classical electron radius, $S_l$ the laser incident light Poynting vector, $\theta$ the scattering angle, $\epsilon = (\lambda - \lambda_l)/\lambda_l$ is the normalized wavelength shift, $\alpha = m_e c^2/(2T_e)$ is the normalized inverse temperature and $S(\epsilon, \theta, \alpha)$, the spectral density function.

The selection of the spectral density function is defined by the ratio between the Debye length and the incident wave frequency. This ratio determines if the scattering from other electrons in the Debye sphere adds incoherently or coherently; in the case of the Thomson scattering, this is an incoherent process. Given that the plasma temperatures in W7-X discharges are high enough to require a relativistic correction, the selected model for the spectral density function must include these considerations.

The spectral density function describes how much power is scattered with a specific frequency and scattering angle by a single electron, given the electron's velocity and the incident wave frequency. In the widely adopted model developed by Naito [58],

$$S(\epsilon, \theta, \alpha) = S_z(\epsilon, \theta, \alpha) q(\epsilon, \theta, \alpha) \tag{4.2}$$

the spectral density function is composed by the Zhuralev coefficient, $S_z(\epsilon, \theta, \alpha)$, and a depolarization factor that considers the incident and scattered light polarization, $q(\epsilon, \theta, \alpha)$. These factors are defined as

$$S_z(\epsilon, \theta, \alpha) = \frac{e^{-2\alpha(x-1)}}{2K_2^*(1+\epsilon)^3\sqrt{2[1-\cos(\theta)](1+\epsilon)+\epsilon^2}} \tag{4.3}$$

$$K_2^*(2\alpha) \approx \sqrt{\frac{\pi}{2(2\alpha)}}\left(1 + \frac{15}{8(2\alpha)} + \frac{105}{128(2\alpha)^2} + \frac{315}{1024(2\alpha)^3}\right) \tag{4.4}$$

$$q(\epsilon, \theta, \alpha) = 1 - 4\eta\zeta\left(\frac{p_0(\zeta) + p_1(\zeta)\eta + p_2(\zeta)\eta^2}{q_0(\zeta) + q_1(\zeta)\eta + q_2(\zeta)\eta^2}\right) \tag{4.5}$$

$$u = \frac{\sin\theta}{1-\cos\theta}, \qquad x = \sqrt{1 + \frac{\epsilon^2}{2[1-\cos(\theta)](1+\epsilon)}} \tag{4.6}$$

$$y = \frac{1}{\sqrt{x^2+u^2}}, \quad \eta = \frac{y}{2\alpha}, \quad \zeta = xy, \quad \alpha = \frac{m_e c^2}{T_e}, \quad \epsilon = \frac{\lambda_s - \lambda_i}{\lambda_i} \tag{4.7}$$

where $K_2^*(2\alpha)$ is the modified Bessel function of the second kind.

Another important factor is that the spectral density function is expressed with a change of variable from $\lambda_s$ to $\epsilon$, that is the normalized wavelength shift between the scattered wavelength $\lambda_s$ and the incident $\lambda_i$.

Regarding the coefficients of the depolarization factor, Naito's suggested (2,2) a rational function approximation to the integral over the scattering angle. Here the depolarization factor's coefficients are

$$\begin{aligned} q_0 &= p_0 = 4 + 30\zeta^2 - 55\zeta^4, \\ q_1 &= 25\zeta^3(29 - 42\zeta^2), \\ p_1 &= -\zeta(25 - 545\zeta^2 + 720\zeta^4), \\ q_2 &= 5(18 - 66\zeta^2 + 630\zeta^4 - 805\zeta^6), \\ p_2 &= 2(33 - 165\zeta^2 + 240\zeta^4 - 100\zeta^6), \end{aligned} \tag{4.8}$$

with the $\zeta$ value calculated in eq. (4.7).

If we reformulate eq. (4.1) to yield a number of scattered photons for a specific channel and spectral channel, we obtain

$$\frac{\mathrm{d}^4 N_s}{\mathrm{d}\Omega_s \mathrm{d}\epsilon \mathrm{d}^3 r} = r_e^2 n_e \langle S_l \rangle S(\epsilon, \theta, \alpha) \frac{\lambda_s}{hc}, \tag{4.9}$$

where h is Planck's constant and c the speed of light. By re-normalizing the wavelength shift we can express the section 4.2 as

$$\frac{\mathrm{d}^4 N_s}{\mathrm{d}\Omega_s \mathrm{d}\lambda_s \mathrm{d}^3 r} = n_e \langle S_l \rangle S(\epsilon, \theta, \alpha) \frac{\lambda_s}{\lambda_l} \frac{r_e^2}{hc}. \tag{4.10}$$

This allows us to arrange the equation to express the number of scattered photons as $N_{ij}$ in the $i-th$ scattering volume, $j-th$ spectral channel

$$N_{ij} = \iiiint n_e \langle S_l \rangle S(\epsilon, \theta, \alpha) \frac{\lambda_s}{\lambda_l} \frac{re^2}{hc} T_{ij}(\lambda_s) \, \mathrm{d}\Omega \, \mathrm{d}^3 r \, \mathrm{d}\lambda_s \, \mathrm{d}t. \tag{4.11}$$

Here we include a calibration factor $T_{ij}(\lambda_s)$ representing the optics transmission coefficient

for the $i-th$ spatial channel and the $j-th$ spectral channel at a specific wavelength $\lambda_s$. Also, due to the nature and possibilities of the calibration process, this calibration factor also includes electronics calibration factors.

The complexity of the model and the physics problem, require a number of assumptions. Initially, Naito already assumed a relativistic Maxwellian velocity distribution function and laser light polarization. Besides this, we can separate the $d^3r$ integral into a surface integral over the laser waist area and a line integral along its path. The small Thomson scattering cross-section ($\sigma_T \approx 6.65 \cdot 10^{-29} m^2$) compared to the laser waist area lets us safely assume that all scattering volumes will experience the same laser power. Knowing that the laser power does not change along the path, we can group $\iint \langle S_l \rangle \, dA$ to substitute it for a factor of laser energy $E_l$ and remove the time integral as well.

Finally, we also know that the solid angle covered by the optics, does not depend on the integration volume, which allows us to express eq. (4.11) as

$$N_{ij} = \frac{r_e^2}{hc} E_l \Delta\Omega_j \iint n_e \langle S_l \rangle S(\epsilon, \theta, \alpha) \frac{\lambda_s}{\lambda_l} T_{ij}(\lambda_s) \, \mathrm{d}l \, \mathrm{d}\lambda_s. \tag{4.12}$$

The last two important modifications to the model that we have to consider are related to the volume length and the actual signal measured by the avalanche photo-diodes. The scattering volume is defined as the area of the laser beam cross section times the selected volume length. If this volume is small enough, we can expect that the plasma parameters within the volume are constant. This allows us to change the integration over the volume length $\int dL$ to a multiplication by the actual length $\Delta L$. Also, due to the fact that the avalanche photo-diode records an intensity over time, the number of photons in this signal is equal to the integral of this signal over time.

$$\left( \int s dt \right)_{ij} = \frac{r_e^2}{hc} E_l \Delta\Omega_j \int n_e \langle S_l \rangle S(\epsilon, \theta, \alpha) \frac{\lambda_s}{\lambda_l} T_{ij}(\lambda_s) \, \mathrm{d}\lambda_s. \tag{4.13}$$

This leaves us with the final required forward model for the i-th spatial channel and the j-th spectral channel to be used in the Bayesian analysis. The complete set of equations are presented here mainly to illustrate the sheer volume of calculations required to yield a single forward model value, and thus the challenge in accelerating such models. Also, the complexity of the model requires the introduction of some assumptions increasing the motivation to use Bayesian Analysis and reduce the number of approximations to a minimum.

## 4.3. Thomson Scattering Bayesian Model

After having a model that describes the involved physics and diagnostic operation, a Bayesian model has to be developed analyzing the relation between the data and the posterior according to Bayes' theorem.

In this case, the two parameters of interest per Thomson channel are the electron density and electron temperature needed to determine the respective spatial profiles. Interferometry adds no free parameters because its density data is predicted by integrating the 10 density values that the TS channels already require. This raises the number of parameters and translates to a 20 dimensional posterior.

Analyzing both models together means that the data has to be mathematically related in order to do a joint analysis. As previously mentioned, these two diagnostics do not only share the density parameter, but also their laser paths are close enough to be considered identical for the practical purpose of this analysis. As seen in fig. 3.1 and fig. 4.1, the overlap of the beam paths of the Thomson and the interferometry laser is a design feature of both diagnostics that eases the comparison of their measurements.

The deviations between their laser paths are due to the angles with which they are directed into the plasma vessel. Design constraints regarding the entry port forces the incidence angle to be slightly different and creates a separation of the beams in the toroidal direction on the order of a few centimeters. In fig. 3.1 and fig. 4.1 this deviation is shown in the poloidal angle for illustrative purposes. This difference has a negligible effect for two reasons. First, the density along a magnetic surface is assumed to be constant and both lasers go through the same magnetic surfaces. Also, they both cross the magnetic axis, where the constant density assumption no longer holds, at the same point. Second, given the current uncertainty and calibration of both diagnostics, changes in density due to this discrepancy are expected to fall within the error margins of each diagnostic. Furthermore, if necessary, one can account for this discrepancy and eventual systematic error in the uncertainty used for the corresponding laser pointing vector parameter.

Based on the assumption of identical laser paths, the line integrated electron density measured by the DI is equal to the integration of the density values for each volume of the Thomson system along the laser path. This allows is to mathematically relate both diagnostics for the Bayesian model development. The joint posterior PDF can be expressed as

$$p(\vec{n_e}, \vec{T_e} \mid \vec{D}_{TS}, D_{DI}) = \frac{p(\vec{D}_{TS}, D_{DI} \mid \vec{n_e}, \vec{T_e}) p(\vec{n_e}) p(\vec{T_e})}{p(\vec{D_{TS}}, D_{DI})}, \tag{4.14}$$

where $\vec{D}_{TS}$ is the data vector from all TS channels, $D_{DI}$, the data from the interferometer, $p(\vec{n_e}, \vec{T_e} \mid \vec{D}_{TS}, D_{DI})$, is the joint posterior PDF. The likelihood, $p(\vec{D}_{TS}, D_{DI} \mid \vec{n_e}, \vec{T_e})$, is defined by a normal distribution of the data set around the predicted forward model values. Finally, $p(\vec{n_e})$ and $p(\vec{T_e})$ are the priors of the selected parameters and $p(\vec{D}_{TS}, D_{DI})$ is the evidence. The evidence factor is disregarded, just like in chapter 3, due to fact that it is constant for a given data set.

As seen in the likelihood function, the interferometer data $D_{DI}$ is not treated as a vector but instead as a single data point. In the chosen the Bayesian model, interferometry data is modeled as if the interferometer provided line integrated electron density as a signal and not its typical raw data. Complications with the storage of raw data of the interferometer for the first operation campaign prevented the availability of a test with realistic raw data. Nevertheless, the processed data for line integrated electron density was available for discharges where the Thomson data was as well. Given that in the first part of the project, the model and analysis for the interferometer are tested and validated, the joint analysis is done using averaged processed data from the interferometer. An alternative could have been to create synthetic data when the forward models are available. While this is useful option, it is crucial to test the models and the analysis under realistic conditions using real data from W7-X.

Moreover, the different temporal resolutions of the two diagnostics in the analysis have to be taken into account. The interferometer provides a 50 Msps data rate which results in 1000 samples within the period of its 50 kHz frequency. The Thomson system on the other hand, provides a single measurement every 10 Hz. Since they are not triggered simultaneously, the time point of the samples will rarely match and different densities may be reflected on their data. Formally, the possible variations of density within two interferometry time points should be introduced in the model which would in turn increase the uncertainty of the marginal posterior for density. Nevertheless, given that events changing density in that timescale are very rare, the typical increase in uncertainty from a change is negligible compared to uncertainty already present in the analysis. For this reason, a window of two data from the interferometer is taken around each Thomson time point. The window of DI samples is averaged and a mean value is used.

For the purposes of this approach, it can be seen as if the DI diagnostic provided the line integrated electron density directly as raw data. The consequences of this assumption and its lack of impact in the acceleration is nevertheless duly discussed in the conclusion section.

Having defined how the data is received from each diagnostic, we can select a formulation for

the likelihood with data coming from different sources, which can be written as [52]

$$p(\vec{D}_{TS}, D_{DI} \mid \vec{n_e}, \vec{T_e}) = \mathcal{N}(\vec{D}_{TS}; \vec{V}_{TS}, \vec{\sigma}^2_{TS}) \mathcal{N}(D_{DI}; V_{DI}, \sigma^2_{DI}). \tag{4.15}$$

For this joint likelihood analysis, the normal distributions, $\mathcal{N}$, of the data $D_{DI/TS}$ over the predicted forward modeled value $V_{DI/TS}$, a function of $n_e$ and $T_e$, are multiplied. Consequently, $\sigma^2_{TS/DI}$ is the covariance matrix and variance value of the TS and DI data respectively, representing the noise level of the data. This results in a 20 dimensional posterior PDF and a covariance matrix that is used to create a suitable proposal distribution for the MCMC algorithm.

Finally, the selection of the model for the prior is made. For the density limits in eq. (4.16), the planned heating scheme is responsible for setting a boundary. The density is controlled to prevent the point where the heating would be reflected by the plasma and thus, no higher densities were attempted in OP1.1. For the temperature in eq. (4.17), a similar reason is present since the limits of temperature are the ones planned for the operation campaign. Therefore, a truncated normal distribution is selected and specially helpful in cases like the plasma edge where the density is known to be low. This can be expressed for density and the temperature as

$$p(n_e) = \begin{cases} \dfrac{1}{A\sigma_{n_e}\sqrt{2\pi}} \exp -\dfrac{1}{2}\left(\dfrac{n_e - \mu_{n_e}}{\sigma_{n_e}}\right)^2 & 0 \leq n_e \leq 1 \times 10^{20}\,\mathrm{m}^{-3} \\ 0 & \textit{otherwise.} \end{cases} \tag{4.16}$$

$$p(T_e) = \begin{cases} \dfrac{1}{A\sigma_{T_e}\sqrt{2\pi}} \exp -\dfrac{1}{2}\left(\dfrac{T_e - \mu_{T_e}}{\sigma_{T_e}}\right)^2 & 0 \leq T_e \leq 20 \times 10^3\,\mathrm{eV} \\ 0 & \textit{otherwise.} \end{cases} \tag{4.17}$$

Here, $A$ is the normalization value for the truncated normal and $\mu_{ne_i}$ the mean value for the density value of each specific channel that requires a different truncation.

Once the Bayesian model has been defined, the next step is to test the model with data from both diagnostics. The following section explains the software implementation of the model, the algorithms chosen to carry out the inversion and the results obtained.

## 4.4. Software Implementation

One of the main reasons why Bayesian analysis is chosen for the estimation of plasma parameters is the capability to handle and rigorously evaluate of the uncertainty associated with the selected parameters. In chapter 3, the estimation of the uncertainty in a one dimensional posterior is reduced to a simple calculation of the standard deviation in the posterior distribution.

When dealing with higher dimensional posteriors the search for the most likely value and its uncertainty can become a more challenging task. As seen in section 2.3 there are several algorithms that are often used for these two tasks.

The posterior distribution provides a full description of the parameters given the data. This means that an exploration of the posterior can be done with a tool like MCMC by taking representative samples of the distribution. These samples, when viewed as a histogram for each parameter value, provide not only the most likely solution but an uncertainty for each of the parameters. This comes at the cost of a longer time required by the iterative algorithms to find a good starting point for the MCMC chains. A selection and tuning of these algorithms is necessary to find a simple way to scan the posterior without incurring too long processing times. This is why a full software implementation and optimization of the analysis is needed before attempting the hardware acceleration.

### 4.4.1. Algorithm selection

The MCMC algorithm, similar to many sampling algorithms, requires a number of iterations on the distribution as a *burn in* period. This allows the chain to enter a high probability region (the vicinity of the maximum in the distribution) from which the representative samples are taken.

Since the main objective of this work is acceleration, any reduction of iterations is beneficial. The reduction of *burn in* iterations can be achieved by starting the MCMC chain at a point close to the distribution maximum for each free parameter. For most real cases, if one chooses an arbitrary starting point, MCMC will spend a considerable amount of iterations in a region of negligible probability. By starting at a high probability point, the time MCMC needs to arrive at a significant probability region can be reduced. The obtained samples in a high probability region are considered representative and can used to estimate the mean and standard deviation of each parameter.

From the available optimizers, the one selected for a 20 dimensional posterior was Hooke and Jeeves. Besides having the robustness of a pattern search algorithm, it was also available as an

inversion tool in Minerva which simplified testing.

### 4.4.2. Testing scenarios

Section 4.3 argues that for a proper testing of the analysis, regardless of the acceleration possibilities, the current model is best tested with real W7-X data rather than using synthetic data. This requires small adjustments to adapt to the data availability. Besides the corrections needed to use a line integrated electron density signal instead of raw interferometry data, another factor to take into account is the selection of data.

The data used in this project pertains to the first operation campaign of the W7-X, where hundreds of discharges were carried out. These discharges vary in length, temperature, configuration of the magnetic field and many other parameters. Besides these variations, on each operation day the diagnostics can have a diversity of behaviors that depend on the conditions of their active elements. The reasons for these differences can be caused by many factors ranging from temperatures and calibrations of the diagnostic's parts, to the global physics configuration of the W7-X that changes typically after several discharges.

This means that the estimation of the parameter they share, density in this case, can yield different results when comparing the diagnostics. While in some experimental scenarios they can agree, in others the estimation of the density can be very different. The main reason for this is that the model does not cover all the previously mentioned variations the diagnostic can have. Whether that is because the model is intentionally simplified or due to a lack of knowledge of the inaccuracy of the model is irrelevant. Finding and solving these inconsistencies is not the objective of this work, yet it is important to cover the possible range of cases.

The punctual density measurements of TS profile can be used to calculate a line integrated electron density, which in turn can be compared against the DI results. Therefore, to cover the range of scenarios cases are compared. In the first case the results of both diagnostics agreed, in the second they show different density estimations. Both belong to a set of scenarios that were previously compared after the experimental campaign [59].

### 4.4.3. Software analysis and results with Minerva

When dealing with real-time data, one cannot predict whether the analyzed scenario will be one where results from both diagnostics agree or disagree. So regardless of the different behavior of the data in both scenarios, a common analysis has to be conducted for both cases.

For the determination of uncertainties and the most likely values, the Markov Chain Monte

Carlo algorithm was selected. A starting point for the MCMC chain is obtained with Hooke and Jeeves to reduce the number of iterations.

**Analysis and fine tuning of inversion algorithms**

From these algorithms, the main factors that have to be tuned and optimized are: the number of iterations and the proposal distribution for MCMC as well as the initial iterations needed with Hooke and Jeeves for a proper starting point. In order to tune these values, different data plots are analyzed under a variety of tuning conditions.

Regarding the tuning of MCMC, the general question of how many iterations are needed until the samples represent the real distribution is a very complex one. Eahc sample is an iteration and samples are collected until the chain is considered to have explored the distribution. This is often referred to as *chain convergence* of the MCMC algorithm, and represents the quality of approximation of the collected samples to the explored distribution. The convergence does not have a direct numerical solution and two approaches are typically followed. The first is through numerical algorithms, used to estimate chain convergence, including **chain convergence diagnostics**. These are often used for post processing of complex problems with little data and intractable posteriors [60]. The second, which was used in this work, is for problems where the parameter's behavior is better known. These are more pragmatic methods, like observing sample acceptance rate and chain behavior. This helps estimate when a chain is believed to be in a higher probability region. For both cases, it is never absolutely certain whether the chain is in a high probability region due to the nature of MCMC and the fact that a perfect reconstruction of the posterior would only be achieved after infinite iterations.

For reasons explained in the following sections, the MCMC algorithm was only used in the software implementation. Nevertheless, keeping acceleration in mind, the number of iterations should be reduced to a minimum. After the reduction, enough samples have to be taken in order to consider them representative of the distribution.

There are several ways to analyze the data in order to estimate an adequate number of iterations. The first is a plot of the samples taken with each chain. A chain with drastic drifts is unlikely to have found a high probability region and thus drifts along the distribution. This analysis is also accompanied by monitoring the acceptance rate to evaluate the chain convergence [61].

Another plot is a histogram of the samples taken by MCMC after the burn in period. These are plotted per channel for density and temperature in order to observe the marginalized distributions for each parameter and channel. From the histogram the mean and the standard

deviation for each parameter can also be obtained and used to plot the profile with the respective channel positions. This can be compared against previous knowledge or expectations of parameter values to determine a chain's behavior as done in section 4.4.3. An example of this is present in appendix A.3.

Regarding the second factor to optimize, the proposal distribution, its selection is key to the behavior of the algorithm and chain mixing. The selection of a good proposal distribution directly affects the number of samples required to properly represent the distribution and can be carried out through several approaches. These range from manual adjustment in a post processing scheme, to an "on the fly" modification by using Adaptive MCMC [62].

If the objective is a real-time scenario, the adaptive proposal is the option that better suits the processing scheme. Nevertheless, for reasons that will be explained in section 4.5, this work will focus on the acceleration of the forward model. A hardware implementation of the adaptive proposal distribution was not found and was not implemented in this thesis. This means that a fixed proposal distribution is the most realistic solution for an FPGA implementation. The behavior of a fixed proposal is therefore tested in the software implementation. The solution suggested and tested in this work is the construction of a suitable proposal distribution, based on previously analyzed plasma discharges.

With the use of the adaptive proposal algorithm implemented in Minerva, a proposal distribution is generated using data from a discharge that represents common temperature and density profiles. This proposal distribution is then used for the other data sets from different discharges and can be used in a real-time scenario. This naturally introduces doubts such as which data to use as a representative case, or whether very different discharges will be affected by a proposal distribution that was not adapted to their posterior. While the absolute values of density and temperature could change from one discharge to another, their correlation depends mainly on the profile shape. In practice, there are only a few profiles that are possible for the analyzed discharges. As a result, the correlation between the Thomson channels will not change drastically and thus their optimal proposal distributions are not significantly different.

Because of the way MCMC's works, it is expected that a sub-optimal proposal distribution would increase the iterations needed to reach a representative distribution. If the number of iterations is fixed for all discharges, the effect of a sub-optimal proposal distribution would be visible in the obtained samples potentially misrepresenting the posterior distribution. If infinite samples could be taken, the sub-optimal proposal distribution would have no effect. For this reason, the choice of the number of iterations used for this joint model is based on the fixed proposal distribution. It is defined by how many samples it requires to lose the bias from using

a sub-optimal proposal distribution.

The last part to be tuned is the number of iterations that are required from Hooke and Jeeves to obtain a good initial estimation that results in a lower amount of burn-in samples. The tuning was done by running the algorithm on synthetic data as well as on plasma discharges where the MAP had already been found. For these discharges, the algorithm was ran until nearly all were within the error margins of the previously estimated MAP. In some cases, the lack of signal in one of the channels can cause a bad MAP estimation for the outermost channel but gives a right one for the rest.

Finally, the last factor to consider is the correlation between temperature and density. While not specifically important for tuning, their correlation is relevant for the validation of the analysis. The TS model depends on both temperature and density while interferometry only on density. The addition of interferometer data, which is a line integration along the TS density spatial channels, constrains the possible density combinations these channels may have. It is therefore expected that constraining each spatial channel's density with respect to the other channels, will also constrain each channel's temperature, thus modifying the uncertainty in the likelihood. In order to visualize this, contour plots are used to observe the correlation between them for each channel as shown in fig. 4.4. This plot shows the shape of the posterior distribution when marginalizing with an MCMC chain for the first channel.

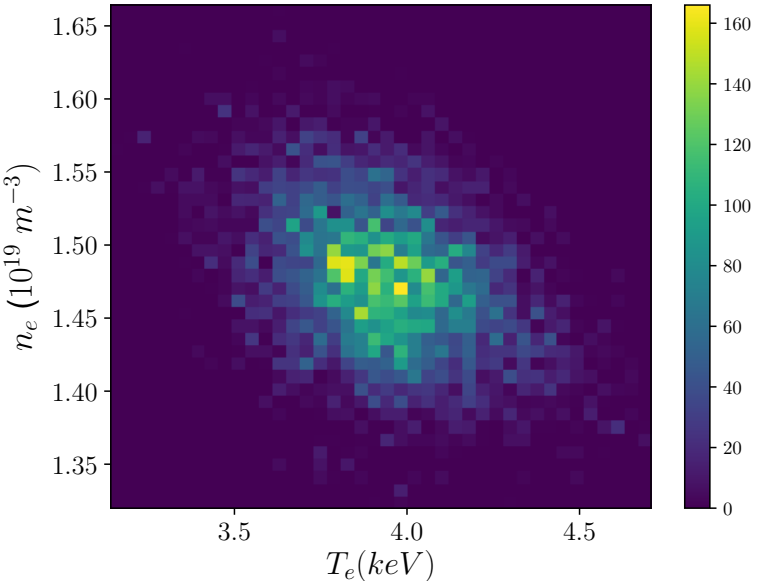



Figure 4.4.: Typical correlation on a 2D Histogram created using the MCMC chain samples for density and temperature of the innermost spatial channel.

Figure 4.4 shows a 2D histogram representing the marginalized posterior distribution for the

innermost channel, close to the core of the plasma. It depicts a soft correlation between density and temperature with a single maximum and represents a typical behavior of correlation seen on plasma discharges where the diagnostics analysis agrees.

In other cases, a low density at the edge of the plasma causes low scattering of light and thus little information in the measurements. This could result in a very widespread distribution indicating that the low amount of information makes it impossible to infer a value with certainty and would affect the channel uncertainty in the profile. By analyzing the data with the described methods, the behavior of the code and algorithms can be tuned to reach an optimal point for the inversion algorithms to run in the minimal possible time.

**Tuning results**

The minimum number of optimizer iterations that yield a high probability starting position of the chain was found to be 10 runs of Hooke and Jeeves. This provided an initial point close enough to the real MAP where the MCMC chains were started to determine a minimum number of sampler iterations.

For the MCMC algorithm, various combinations of *burn in* and representative samples were tested by observing the performance of the chain on each parameter for both representative cases. Once the chain behavior and acceptance rate are deemed sufficient to define that the chain is in a high probability region, the subsequent samples are considered representative of the distribution and used to plot the profile.

This pragmatic task of individually looking at MCMC chains to find a proper amount of iterations, was tested in several steps ranging from 5000 to 500000 iterations. The minimum number of iterations that could properly represent the tested posterior distributions was 40000 . This was done using 20000 iterations as *burn in* and the other 20000 to determine the mean and standard deviation estimation.

With the tuning defined, the stored representative samples of each channel are used to find a most likely value and its standard deviation. Those obtained values are plotted to generate the final profiles of temperature and density as shown in the following two representative cases.

**Cases with good previous agreement between the individual density estimations of TS and DI**

The case where the individual analyses of TS and DI yielded similar density values was the first to be considered as a representative case for the joint analysis. This means that the data of the Thomson channels was analyzed individually and integrated to compare against the line integrated electron density previously analyzed from interferometry.

The analysis with the final tuning resulted in a density profile as seen in fig. 4.5 and a temperature profile shown in fig. 4.6. The shown profiles contain superposed plots from three different analysis: the values of the MAP used as starting points for the MCMC algorithm, the values from an analysis with only TS data considered and the joint analysis done in this thesis. The density profile shows how in the case of a good previous agreement between both diagnostics, the joint analysis also coincides with the TS results obtained in other works. The results using only TS analysis fall within the uncertainty range of the joint analysis, suggesting that the addition of the DI information modifies and increases the uncertainty that was obtained previously.

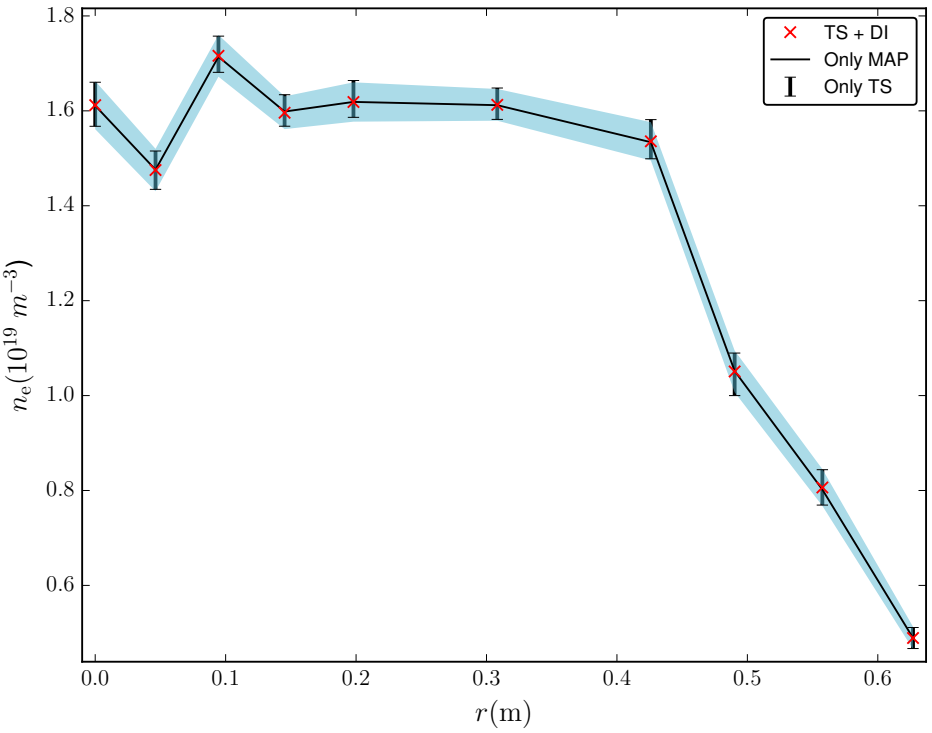



Figure 4.5.: Density profile for good previous agreement with joint analysis values (red x and blue shade) as well as only TS values (black error bars) and the MAP value obtained with Hooke and Jeeves (black profile)

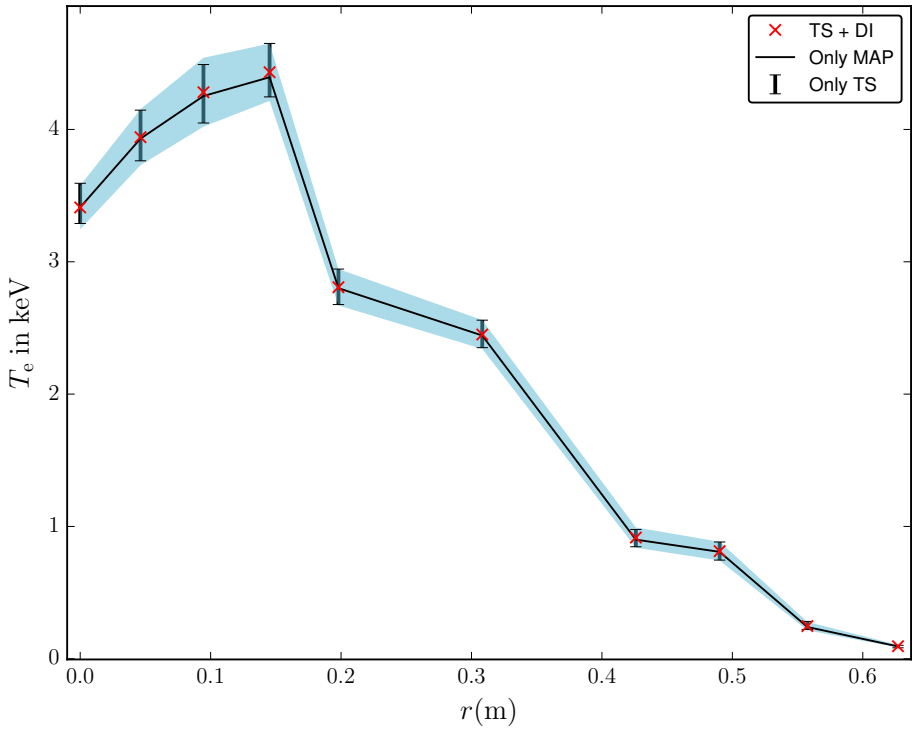



Figure 4.6.: Temperature profile for good previous agreement with joint analysis values (red x and blue shade) as well as only TS values (black error bars) and the MAP value obtained with Hooke and Jeeves (black profile)

Even though it falls outside the scope of this work, it is important to mention that considering the error bars, this density plot does not represent evidence for a hollow profile.

The temperature profile, similar to the density, shows the TS results within the error margin of the joint analysis and a similar the uncertainty from both diagnostics. This shows how the profile is expected to look like and how the forward model, as well as the inversion, agrees with previously analyzed estimations of Thomson profiles.

Regarding acceleration, this scenario presents no complications for the implementation of the FPGA version since it creates no physically impossible profiles. The values of temperature and density all fall within a physically plausible range, opposite to the following disagreeing case where the independent of TS yields a different profile than the joint analysis.

**Cases with bad previous agreement between the individual density estimations of TS and DI**

A different behavior is observed when the line integration of the TS channels results in a different density value to what is obtained with the DI. In the corresponding W7-X discharges, the interferometer data were noisy yet the values of line integrated electron densities appeared to be within a plausible range. The TS diagnostic on the other hand, had problems with the outermost spatial channel which only measured noise. The consequences were visible in the low density values ($< 1 \cdot 10^{17} m^{-3}$) in fig. 4.7 as well as the high uncertainty and erratic estimation of temperature for those channels in the profile shown in fig. 4.8.

The density profile shows how the outermost Thomson channels drops out of the joint analysis error bar due to low information content in the data due to low density and poor scattering.

The reasons why the signal was so low that the two outermost channels are in principle not important for this work. However, the identification of such scenarios is crucial for a general acceleration of this analysis, considering that this profile could be used for plasma control or device safety.

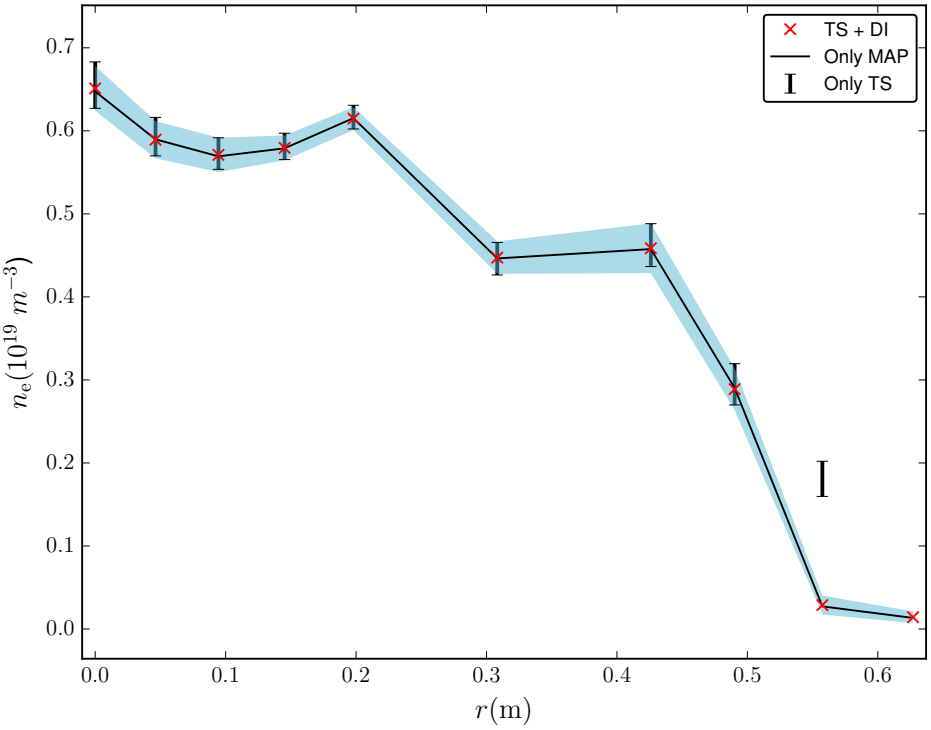



Figure 4.7.: Density profile for poor previous agreement with joint analysis values (red x and blue shade) as well as only TS values (black error bars) and the MAP value obtained with Hooke and Jeeves (black profile)

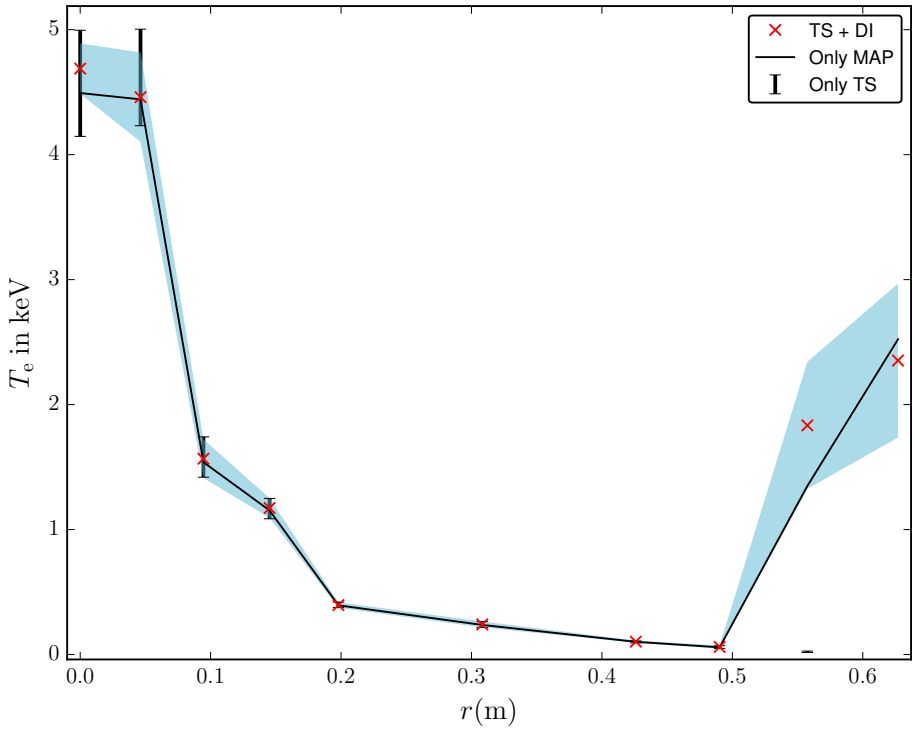



Figure 4.8.: Temperature profile for poor previous agreement with joint analysis values (red x and blue shade) as well as only TS values (black error bars) and the MAP value obtained with Hooke and Jeeves (black profile)

Under these circumstances, solutions ranging from raising a flag to avoiding such misleading results can be attempted. Therefore, a set of reasons behind this implausible result and its alternative solutions are briefly discussed.

The first reason comes from the information content in the temperature profile in fig. 4.8. The temperature values in the two mentioned channels are considerably higher than what is probable for the edge of the plasma. Further inspection showed that the data for these two outermost channels is truncated to zero due to its statistically insignificance, but had a rather large uncertainty (orders of magnitude bigger than the mean). The causal relationship between a high temperature and a low signal level is not intuitive and will be explained.

As mentioned in the introduction of section 4.2, besides the Thomson scattering radiation, background radiation is also generated by brehmsstralung. The small scattering cross-section ($\sigma_T \approx 6.65 \cdot 10^{-29} m^2$) means that when the spatial channels measure a very low density, a strong background radiation is enough to result in a measurement with a very low signal-to-noise ratio. Figure 4.6 shows how under normal conditions, the lowest temperatures are measured by the two outermost channels. Therefore, these spatial channels are the most likely to result in a

noisy signal during low density conditions and explains why the analysis yields implausible results for these two.

To understand the reason for the results giving implausible high temperatures, the spatial channels response under low densities must be analyzed. For the cases where no signal is measured, the polychromator's spectral channel with the broadest filter will be most susceptible to background radiation. Hence, that spectral channel registers the highest noise level and biggest uncertainty of the measurement. If we consider the wavelength range of each polichromator's spectral filter, as shown in fig. 4.3, we can see that filter number 5 is the broadest and covers the range that would detect higher temperatures. Thus, the uncertainty of this spectral channel will be considerably higher. With the discussed effects of low density in the spatial and spectral channels on the Bayesian analysis we can now explain the results.

Even when all the channels provided noise measurements mainly, the spectral channel providing data with the highest uncertainty and lowest signal-to-noise ratio, is the one measuring the highest temperature range. This means the marginalized posterior for the outermost channel is very broad and carries little information. Specially broad towards high temperatures given the discussed effects on the spectral channel covering that range. This means that the $T_e$ MCMC chains sampling those channels will roam around the broad marginalized posterior and will require an additional undefined number of iterations to properly represent it. If ran for infinite samples, the marginalized posterior would most likely reflect a very broad distribution covering the whole temperature range given the lack of information from the data.

All of this means that in fig. 4.8, the uncertainty and range of the results for these spatial channels show the consequences of stopping the MCMC algorithm too soon for those channels. The chains only explore part of a very broad distribution, which in this case was the high temperature range where the MAP gave the starting point. This was also confirmed by observing the respective MCMC chains which roamed around without converging within the predefined limit of MCMC iterations.

The second factor to consider is the effect of adding interferometry data. The addition of interferometry data will limit the number of possible density combinations that the spectral channels can have, in order to satisfy the measured line integrated electron density and its uncertainty. Given a density and temperature correlation in the TS model, it is expected that uncertainty of the marginalized posterior for the outermost channels would be limited by this same principle. However, for the disagreeing plasma discharges in fig. 4.9, we see that the noise present in the interferometry processed signal, also has a high uncertainty which provides no constraining of the posterior. With a higher uncertainty in a channel measuring only noise, its

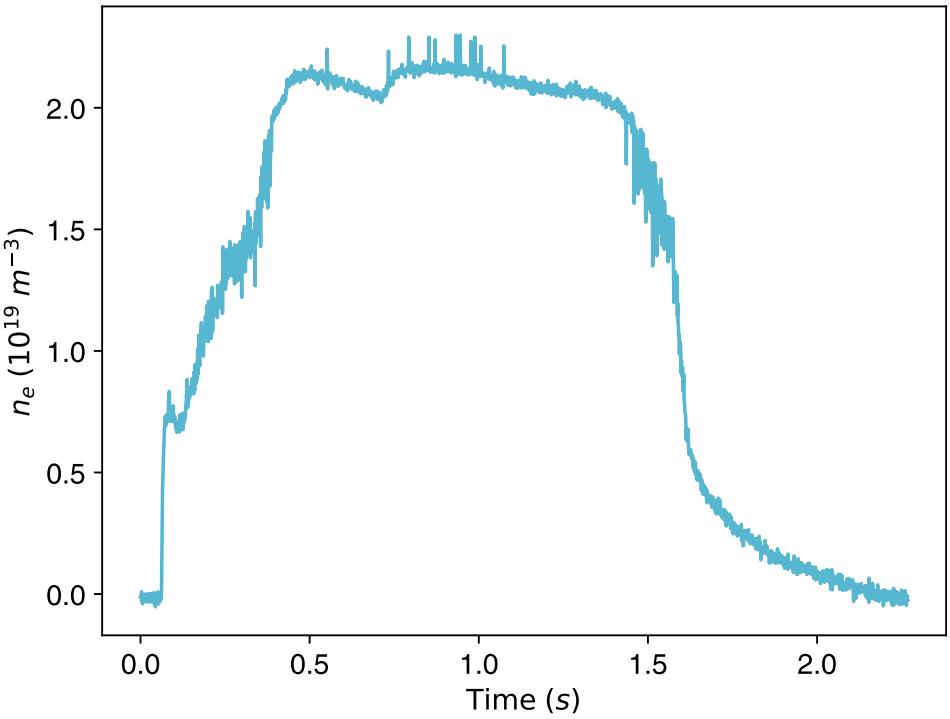



Figure 4.9.: Interferometer's line integrated electron density signal for the poor previous agreement between density and temperature profiles

marginal posterior will be wider and therefore allow the exploration of more combinations of TS parameters.

It is clear that a result like the one shown in these two profiles with such high temperature at the plasma edge is not possible. However, with these two effects combined, the analysis results from the algorithm are expected under the noise conditions of both diagnostics and low signal level of the two outermost channels.

This thesis suggests two solutions to this problem in order to avoid or limit the occurrence of such unrealistic result. With the hardware implementation of the forward model in mind, solutions fitting to the designed implementation will be discussed in the conclusions section.

## 4.5. Hardware Design

The typical way the Thomson scattering analysis is accelerated is through the use of a LUT that has precalculated values for a range of possible input parameters. While this in principle is a very fast solution, the flexibility and other advantages of the Bayesian approach are lost. Moreover, if a parameter needs to be considered as unknown, the assignment of a new probability increases the size and complexity of the LUT and requires a full regeneration of the table. Therefore, this section describes a solution through an accelerated Bayesian analysis.

### 4.5.1. Implications of accelerating a more complex model

When designing a faster joint analysis of interferometry and Thomson scattering, some factors have to be considered that were less relevant in the single DI analysis and its acceleration. The DI acceleration gained from high parallelism of the forward model. This was possible because it dealt with a one dimensional posterior of a relatively small forward model.

For the joint analysis of TS and DI, the models and inversion techniques differ from the single DI model in the several ways. First, the dimensionality of the posterior is much higher than in the first case, now being 20 dimensional. Second, the forward model is more challenging regarding the number of operations or functions it contains, and their complexity and the precision they require. Finally, the inversion technique used to obtain the marginalized parameters and consequently their mean and uncertainty, is no longer achievable by a simple comparator scheme and thus requires iterative algorithms.

The first challenge regarding dimensionality, mainly translates to a bigger resource consumption as well as limiting the inversion techniques that can be used. An increase in free parameters suggests using a parallelization approach. While a small level of parallelism can be implemented for each TS channel, the size of the forward model for each channel must be considered regarding the FPGA resource limits. A parallel approach to evaluate the 3700 wavelength values of the forward model, would be intractable. For this aspect a pipelining approach would be a more suitable solution to increase clock speed and total throughput.

The second challenge is the complexity of the model. It will have a big impact on resource consumption and processing time. Compared to the DI, the TS forward model has considerably more operations, more complex arithmetic factors, and diverse dynamic ranges. This increase in complexity entails an increase in resource consumption and a limitation in how many parallel versions of the forward model can be implemented. Moreover, in order to keep the precision comparable to CPU architecture and satisfy the dynamic range, a double precision floating-point

architecture is advantageous but leads to a bigger resource consumption as well.

A use of this arithmetic precision standard not only simplifies the design of bus sizes by standardizing it, but also moves in the direction of having a generic solution for any model. This helps study the feasibility of FPGA acceleration of Bayesian analysis for different inverse problems. The double precision floating-point architecture would cover the arithmetic precision requirements of most inverse problems. This helps to generalize acceleration approaches and makes it the arithmetic format of choice.

Finally, regarding the limitation of available inversion algorithms, samplers like MCMC have the disadvantage of being iterative. This makes them slower than optimizers and used rather for post-processing. For Bayesian analysis of complex models, these algorithms increase the difficulty of acceleration because the full evaluation of extensive forward models costs significant time before the next iteration can be started. Its sequential nature limits the parallelization possibilities that proved to be crucial for accelerating the DI model. On the other hand, MCMC's ubiquity in most high-dimensionality inverse problems helps attain the goal of a general acceleration design.

After reviewing the implications of accelerating a more complex model, a short analysis of where they can be applied is due. As discussed in section 3.5, the breakdown of acceleration possibilities are the evaluation of the forward model, the application of Bayes' theorem and the inversion algorithm.

An acceleration of a popular sampling algorithm like MCMC to surpass other state-of-the-art accelerations, as the one mentioned in section 2.3.3, could already be the main goal of a project. Also, the application of Bayes' theorem leaves little room for improvement, singling out the forward model as the main candidate. While this sounds like a limitation of acceleration possibilities, it is important to have an overview of what part the analysis requires the most time and how they affect the total processing time.

$$T_{\text{Total}} = (T_{\text{MCMC}} + T_{\text{FM}} + T_{\text{Bayes}})N_{\text{iterations}} \tag{4.18}$$

Equation (4.18) is a simplified model of the total analysis duration when taking into account the main processing requirements of each stage and the typical number of operations each entails. Here $T_{\text{MCMC}}$ represents the inversion algorithm time, $T_{\text{FM}}$ the time required to evaluate a full forward model and $T_{\text{Bayes}}$ the application of Bayes' theorem while $N_{\text{iterations}}$ is the number of iterations MCMC needs.

Figure 4.10 shows the effect of acceleration on each processing time. Each curve is generated

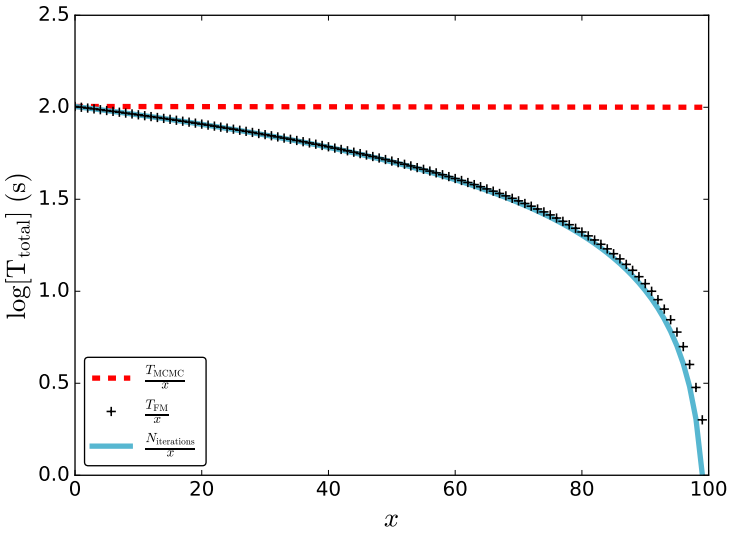



Figure 4.10.: Variation of $T_{\text{Total}}$ in eq. (4.18) when accelerating each time $x$-fold where $1 \leq x \leq 100$. Base values $T_{\text{FM}} = 100\,\mu\text{s}$, $T_{\text{MCMC}} = 1\,\mu\text{s}$, $T_{\text{Bayes}} = 100\,\text{ns}$, $N_{\text{iter}} = 1000000$

by multiplying each individual with of $1/x$ where $1 \leq x \leq 100$ and keeping the rest at their base values. These base values were taken from those obtained on Minerva for the TS model. The effects observed can be explained by the fact that in a model like the one analyzed, the forward model tends to contain tens or hundreds of operations sometimes for several channels. The MCMC algorithm and Bayes' theorem only require a couple of operations and their acceleration has a negligible effect on the total processing time. Thus, the most relevant acceleration segments are the number of iterations and the forward model. Besides this, a complex forward model is often observed in many inverse problems. For these reasons, the acceleration of the forward model is a key element and this section will focus on its acceleration.

### 4.5.2. Architecture Design

While both the TS and DI forward model have to be implemented, the DI model is substantially smaller and was previously tested. For this reason the focus lies on the Thomson scattering model.

#### Exploiting parallelism and pipelining on the FPGA architecture

The initial acceleration possibility for the FPGA design is the parallelization and design of dedicated modules of the model's factors to reduce processing time. The most obvious possibility targets one characteristic of inverse problems in physics. Several data sources from N channels of a single diagnostic can be parallelized. On a global level, the forward model for each channel

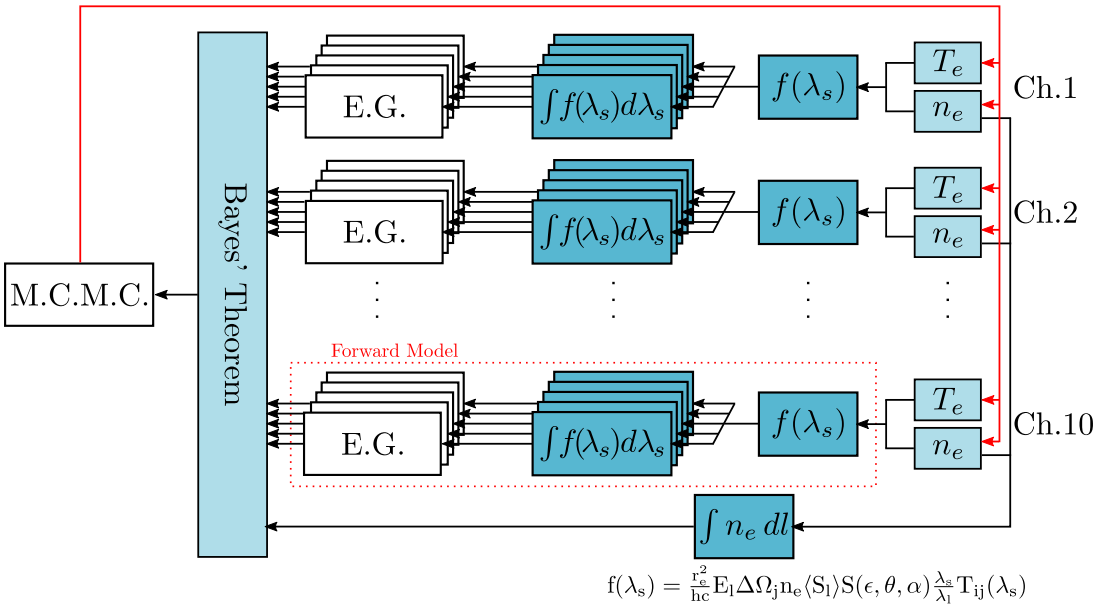



Figure 4.11.: Proposed FPGA full architecture with: partial forward model ($f(\lambda_s)$), spectral integration per spectral channel ($\int f(\lambda_s)\, d\lambda_s$), electronic gains (E.G), Bayes' theorem and inversion (M.C.M.C.) and the DI model ($\int n_e\, dl$)

and the integration for each spectral filter described in eq. (4.13), can be parallelized to reduce processing time. This results in a general design concept shown in fig. 4.11.

Here the registers of the main free parameters, the density and temperature for each channel, are at the base of design. A section of Naito's formula follows ($f(\lambda_s)$) within the forward model, before a parallelized calculation of the 5 spectral channels is carried out. The integrated outputs are then multiplied with their respective electronic gains finishing the forward model. With the prediction of the measured values available, the likelihood is calculated and a prior is applied in the Bayes' theorem module before sending the results to the sampler module. In the M.C.M.C. module, the decision of accepting or rejecting the new sample is taken and the next value is sent to the $T_e$ and $n_e$ registers to finish a full iteration of the analysis.

At a more intricate level, within the forward model acceleration opportunities are clear from the start such as the implementation of eq. (4.2). Here, $S_z(\eta,\theta,\alpha)$ and $q(\eta,\theta,\alpha)$ can be calculated simultaneously. Nevertheless, the sheer volume of operations required and their possible configurations open many other factors that may be suitable for parallelization but not trivial to identify. This is specially visible when dealing with numerous arithmetic factors within the model that do not depend on each other and can be processed simultaneously.

To optimize the time required for each factor in the final model, two main tasks are necessary. The first task is the analysis of how each factor can be formulated in different ways by analytic manipulation of the formulas. Whether that is to factorize or expand the equations, it is done

in order to promote parallelism and reduce the number of operations. The second task is the mixture of parallelism with pipelining (section 2.4.1). The model requires a spectral integration over a range of 3700 wavelength values, which is an ideal case for pipelining the architecture. When the best formulation for parallelizing an expression is found, many possible configurations can be designed by grouping operations and considering dependencies so that the waiting or buffering time of values in the pipeline is reduced.

For models as extensive as the TS case, the factorization and testing of different formulations of the equation in order to optimize parallelism can grow in difficulty. This type of analysis and optimization can be a cumbersome and time-consuming design process. To address this problem systematically, a special software tool (described in section 2.4) was developed. It simplifies the optimization time required to test different configurations and the application of parallelism and pipelining. Moreover, the tool allows for a fast and effective way to test different factorization possibilities and architectures of each term in the model, in order to compare them. It also allows to visualize the whole architecture in order to improve sections where the buffering of a value in the pipeline can be reduced. Finally, the tool's ability to define the processing time of arithmetic operation in terms of clock cycles, simplifies the tracking and programming of buffering lengths in the pipeline. The tool simplifies the visualization of these factors and adjusting their calculation order to reduce buffering resources.

With it, parallelism was exploited in the forward model in order to modularize individual factors and decrease processing times with an FPGA architecture as shown in fig. 4.12, fig. 4.13, fig. 4.14 and fig. 4.15. All of these designs start from right to left, with the base level containing constants, the main free parameters and the beginning of critical path shown in red. The circles represent the operation applied and the squares are the registers where the partial results are stored. This critical path starts with the test wavelength value $\lambda_t$ of the analyzed spectrum as shown in fig. 4.12.

At its maximum, there are 11 parallel branches in the design calculating factors simultaneously, most of them are present in fig. 4.14. Nevertheless, there are numerous more instances in the design that calculate two or more operations at the same time. This is one of the two main reasons for the design's acceleration. The other main factor is the dedicated pipelined hardware implementation of the model. It results in a throughput of one wavelength value per clock cycle once the chain of values has propagated through the entire critical path.

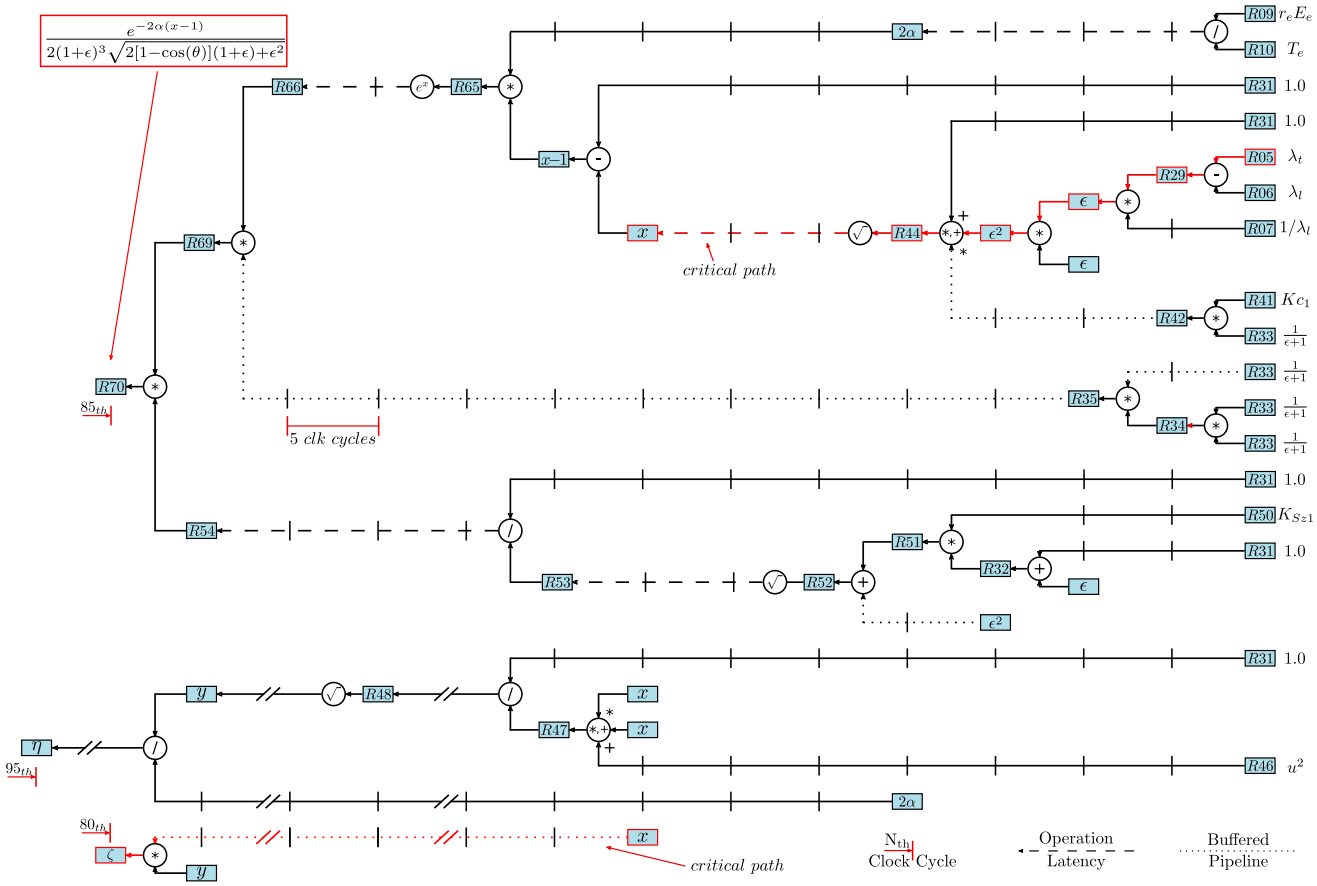



Figure 4.12.: Schematic for Zhuralev coefficient (eq. (4.3)). The critical path (red) is part of the generation of $x$ in $S_z$ and continues with $\zeta$ in the depolarization factor (fig. 4.14).

### Accelerating while reducing resource consumption

While parallelism and pipelining of 3700 wavelength values introduces already an acceleration, the resource usage has to be kept in mind. As discussed, an arithmetic precision of double-precision floating-point is desired to explore the reach of the FPGA implementation.

Previous FPGA generations would have presented resource limitations as well as poor availability of IP-cores for the application of this arithmetic. As discussed in section 1.3, current FPGA generations not only simplify and pre-optimize the implementation of these complex arithmetic functions with floating-point arithmetic cores, but also posses the resources to use them in extensive models. In the architecture design, all the floating-point operations were done using Xilinx's IP-cores and were optimized for performance. The optimization towards performance, instead of area, for a heavy resource consuming standard like floating-point means that the reduction of the resource consumption has to be achieved somewhere else.

Several changes towards using less resources were also considered in this optimization process. Divisions (3220 slice LUTs, 2066 slice registers), exponentials (2248 slice LUTs, 704 slice registers,

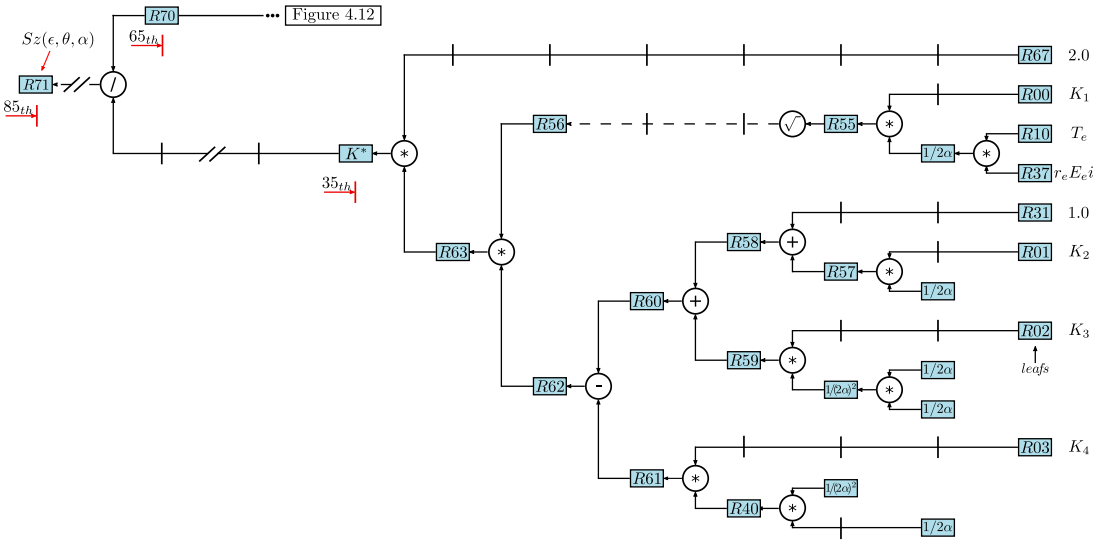



Figure 4.13.: Bessel factor $K^*$ (eq. (4.4)), as a part of the Zhuralev coefficient ($S_z$).

15 DSP slices) and square roots (1701 slice LUTs, 968 slice registers) require considerably more resources than other operations like a multiplication (140 slice LUTs, 147 slice registers, 10 DSP slices) [1].

Many factors of the model can be modified or mathematically manipulated to reduce resource usage, mainly by avoiding costly divisions. Multiplications with the inverse of a constant replace the costly divisions over the same constant. This is shown in the design of the Bessel factor(eq. (4.4)) represented in fig. 4.13, with the constant $K_2$ representing 15/8.

Described in fig. 4.13, is the effect of changing the order the operations to minimize resource consumption. Instead of calculating $2\alpha$ and $(2\alpha)^2$ and dividing over each term, $1/2\alpha$ can be calculated once in order to implement its squared and cubed values with multiplications.

Finally, one of the modifications that was made introduces a trade-off scenario in terms of resource consumption. In order to reduce the critical path length, the factor of $1/(\varepsilon+1)$ was not obtained by using the calculated value of epsilon in fig. 4.12. In the critical path, this value is only available after the tenth clock cycle. Instead, the factor is taken from a table implemented on a ROM with its 3700 values as calculated with eq. (A.4) in appendix A.2. This was not done with the other possible values that could have been precalculated, in order to keep the total number of operations implemented as close as possible to the software approach for comparison purposes. Nevertheless, the specific value of $1/(\varepsilon+1)$ directly affected the critical path and was substituted for acceleration purposes.

The reduction of resource consumption by changing the implemented operations can be problematic for the comparison with the software approach, because it changes the total number

[1] Values based on specific optimization options and design parameters discussed in the following subsection

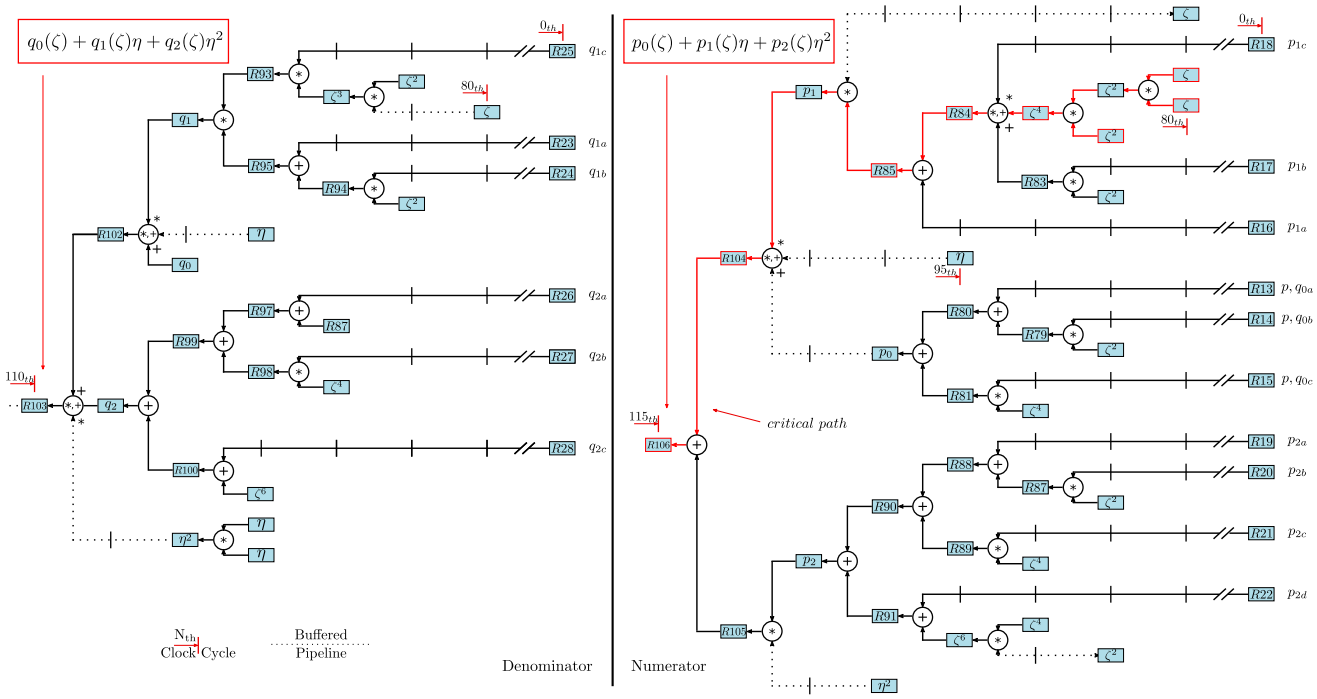



Figure 4.14.: Numerator(right) and denominator(left) of the depolarization factor (eq. (4.5)). The numerator contains part of the critical path (red) coming from the $\zeta$ factor.

of operations. To evaluate this, an overview of the acceleration causes has to be kept in mind. For the FPGA design, the number of operations changed or reduced that were not changed in the software approach was 7 from the total of 87 operations in the FPGA design. These changes were all applied before the spectral integration. Nevertheless, the main driving factor for an acceleration is the high level of parallelism plus pipelining introduced by the FPGA. Parallelism is exploited in the branches of the numerator and denominator in eq. (4.5), as well as all the shared factors eq. (4.7) and eq. (4.6) shown in fig. 4.12. Moreover, the simultaneous calculation of the two main factors $Sz$ and $Q$ show that the acceleration is not driven by the few changes made to 7 operations. Finally, if the pipelining of a dedicated hardware design is taken into account, these changes do not affect the comparison. These changes mainly help to avoid resource consuming operations in a forward model that, due to its volume, already presents a challenge.

### General design considerations

Regarding parameters general to the whole design, an initial test clock frequency of 100 MHz was selected as a single frequency to avoid clock domain crossing on a design that did not demand it. With the base frequency selected, different operation latency were tested for each operator in order to prevent timing violations while minimizing the critical path. To

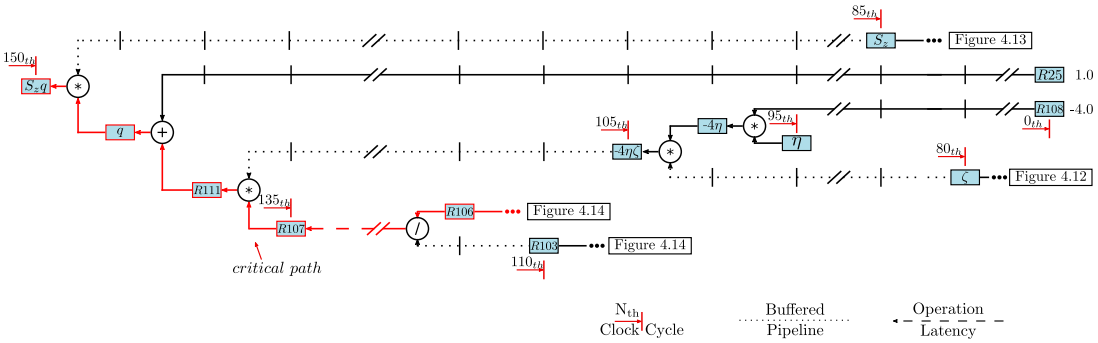



Figure 4.15.: Final branch (eq. (4.5), eq. (4.3)) joining the Zhuralev coefficient $S_z$ (fig. 4.13) and two elements of the depolarization factor $q$ (fig. 4.14) to finish the critical path used in the benchmark.

meet timing requirements and improve pipelining, a latency of 5 clock cycles was selected for simpler operations like sums and multiplications. For the slower counterparts, i.e. divisions, exponentials and square roots, a latency of 20 clock cycles was used.

The design was initially targeted for implementation on a Virtex-7 xc7vx690T FPGA because of its resource abundance. However, this FPGA was not available for testing. Therefore, the design was programmed on the KC705 development board that uses a Kintex-7 xc7k325t FPGA. Performing a simulation and achieving an implementation is in general sufficient to assess the feasibility of the design running on an FPGA. However, the implemented circuit was tested on the Kintex FPGA to certify the lack of timing issues and that the calculated data validity.

## 4.6. Results and analysis

Regarding results pertaining to the architecture design, an initial result is the reduction of the critical path when using a parallel architecture. If FPGA parallelism was not exploited and the operations were carried out sequentially, considering the individual operation latency, the single critical path in $f(\lambda_s)$ shown in fig. 4.11 would take 695 clock cycles. The critical path of the FPGA design starts in fig. 4.12 and ends in fig. 4.15. Using the developed tool for optimizing the design, the critical path length was reduced to 150 clock cycles.

The other results to be analyzed focus on data validity and the total acceleration. In section 3.5 the validation and acceleration of the FPGA implementation of the DI was done at a global level. That is, they were benchmarked for the full analysis of both the hardware and software version. Given the particularities of the current model, both the data validation and acceleration are not done for the full analysis, but only for the most time-consuming sections.

The data validity is to some extent trivial given that both use double-precision floating-

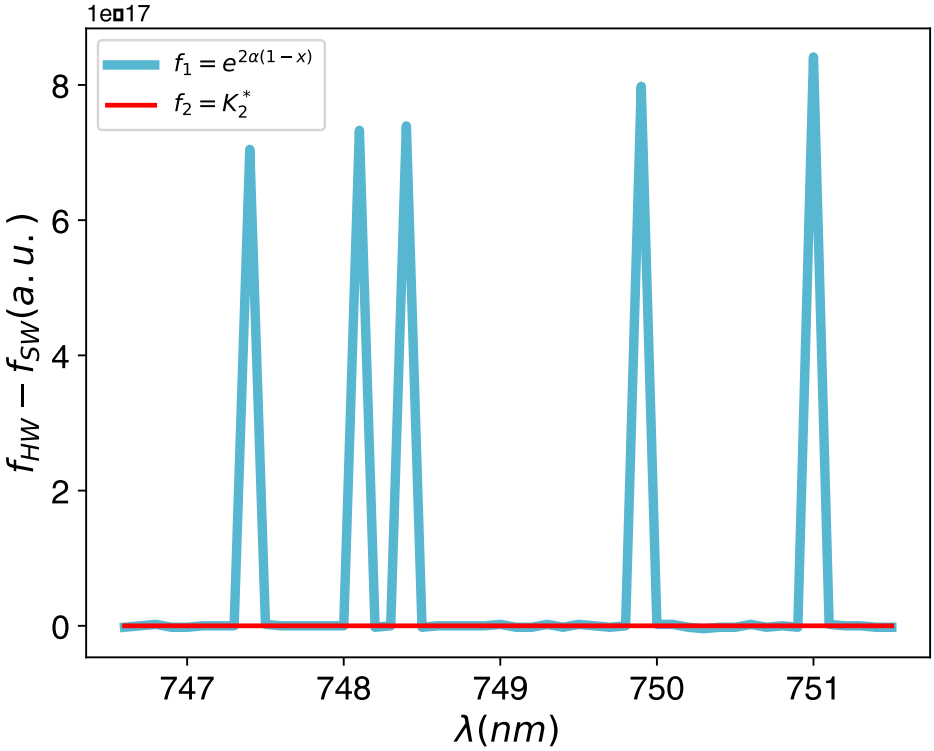



Figure 4.16.: Differences between the software and hardware results. The chosen representative functions are the exponential function $f_1 = e^{2\alpha(1-x)}$ whereas in other functions like $f_2 = K_2^*$ the results match exactly.

point arithmetic. The acceleration, on the other hand, is more complicated to determine. To evaluate acceleration, a benchmarking point had to be chosen to measure a relative acceleration. The possibilities to accelerate the analysis were mentioned to be in the forward model, the application of Bayes' theorem and the inversion algorithm. However, the timing model in eq. (4.18) suggests that focusing the effort on the acceleration of the forward model and iteration reduction is the most reasonable approach. This means that the forward model was accelerated and the inversion algorithm, due to its ubiquity and complexity, was not implemented. For this reason the validation and acceleration is benchmarked using the forward model; specifically the function $f(\lambda_s)$.

Starting with the validation of the forward model results, its precision had an acceptable agreement with those in the Minerva implementation, as is expected of double precision floating-point architecture.

Figure 4.16 shows the comparison of forward modeled values for $e^{2\alpha(1-x)}$ and $K_2^*$ in both software and FPGA implementations. The propagated error in the final $f(\lambda_s)$ originates from the implementation of the exponential function in hardware, which had a small disagreement

Table 4.1.: Acceleration with FPGA based on a CPU time of $(0.8 \pm 0.1)$ µs

| **CPU Freq.(MHz)** | **FPGA(ns)** | **Acceleration(N-fold)** |
|---|---|---|
| 100.0 | 10.40 | 82 |
| 200.0* | 5.20 | 164 |
| 300.0* | 3.46 | 246 |

with the software results. The other functions, as seen in $K_2^*$, completely agree with the software results. These differences are most likely to be from the method used by both tools to evaluate the exponential function. Another error source can be the systematic error introduced by the difference in round of techniques used in Java versus the Xilinx floating-point IP-Cores. However, the accumulated error from the exponential function after the spectral integration amounts to a difference in the order of $10^{-16}$ and has a negligible effect on the final result.

Regarding the comparison of FPGA and the Minerva code processing times, several aspects are important. Developing a full version in C, as was done in chapter 3, requires a workload that is not really justified by the achievable speed-up when compared to the code in Java. While a C code can be faster and more efficient than its Java counterpart, the speed up will be small compared to the orders of magnitude of acceleration achievable with an FPGA. This is visible in the results show in table 4.1. A C implementation would only be relevant if the processing time difference between the FPGA and the software implementation, is small enough to fall within the range of processing time differences that C and Java can have. For this reason the FPGA performance is compared with the performance in Minerva.

The benchmarking is based around the processing time of the model for each wavelength value, a central parameter that is accessible to benchmark in the Java code. Moreover, the total processing time of $f(\lambda_s)$ can also be derived from this base duration.

The Minerva code was run on an Intel Xeon E3-1505M CPU with a clock speed of 2.8 GHz and 16 GB of RAM memory. Visual VM was used for the Minerva profiling on Java, focusing on the application run-time. Using data from 20 runs, the Minerva calculation of the forward model took on average $(0.8 \pm 0.1)$ µs for each wavelength value.

On the FPGA, this is the time a single wavelength takes to go through the critical path. This path starts in fig. 4.12 and ends in fig. 4.15. The total processing time for the FPGA eq. (4.19) is calculated as follows.

$$T_{FPGA} = (L_{\text{critical path}} * \tau_{clk} + (N_{\text{WL}} - 1) * \tau_{clk})/N_{\text{WL}} \tag{4.19}$$

Here $T_{FPGA}$ is the total FPGA processing time for the evaluation of $f(\lambda_s)$ for a single

wavelength value. $L_{\text{critical path}} * \tau_{clk}$ is the length of the critical path multiplied by the clock period. It represents the time the first value of the wavelength takes to propagate through the whole pipeline. $N_{WL}$, represents the number of wavelength values. This means that the second factor $((N_{\text{WL}} - 1) * \tau_{clk})$ is the time required by the rest of the values given the throughput of the pipeline after the initial propagation. The reason for the averaging over $N_{WL}$ is due to the effect of pipelining. The first value requires the full propagation of the wavelengths chain through the whole critical path. After that, the rest of the values only need a single clock cycle to be ready. The critical path of the forward model requires 150 clock cycles. With a wavelength resolution of 3700 values, the average duration for the processing of a single wavelength value is 10.40 ns. These results show in table 4.1, the acceleration achieved with the initial clock frequency and two other projected frequencies that stay within the operation limits of the used IP-Cores. With the designed architecture, an $\approx$ 82-fold acceleration of the forward model is achieved. This already shows the general potential of acceleration in complex inverse models where the number of arithmetic operations is significant.

The projected increase in the clock frequency could require changes in operation latency, given that the timing constrains would be tighter. These projected values of total processing time are done without considering latency changes, which could increase the critical path. However, the effect of a change in the critical path is not so relevant for the processing time if compared to the acceleration caused by doubling or tripling the clock frequency. For example, a change form 150 to 180 stages in the critical path, would yield an acceleration factor of 162 and 244 respectively.

In the Thomson model, further acceleration possibilities are available. Given that this calculation is done per channel, any N-th order parallelization of spatial channels in the forward model would make the analysis N-times faster at a cost of higher resource consumption. A similar possibility exists for the five spectral channels of each polychromator, where the spectral integration along the range of wavelength values can be parallelized as well. In this case the resource consumption would not be as affected.

From the profiling of the Minerva analysis, the base values for fig. 4.10 were obtained, where it was confirmed that the forward model calculation represents the biggest time-consuming task. This was relevant to determine how accurate an approximation of the total processing time would be in a full implementation of the analysis on the FPGA. The FPGA processing times for the evaluation of Bayes formula and MCMC algorithm cannot be accounted, since they were not implemented in this project. However,the processing time of forward model is some orders of magnitude larger than its counterparts. This means we can approximate the

Table 4.2.: Resource Consumption for a Single TS Channel

| **Resource** | **Utilization** | **Available** | **Utilization %** |
|---|---|---|---|
| LUT | 61035 | 433200 | 14.09 |
| FF | 39269 | 866400 | 4.53 |
| LUTRAM | 3143 | 174200 | 1.8 |
| DSP | 721 | 3600 | 20.03 |

FPGA processing time for a full iteration of the analysis to be within the processing time of the forward model for the 3700 wavelength values. If the forward model for each channel were run in parallel at the base frequency of 100 MHz, the 40000 iterations would require $\sim 1.55s$. This brings the usual post-processing technique closer to a real-time frame with a design that has not been fully optimized.

Of course, this level of parallelism prompts the question whether the size of the model stays within resource availability of current modern FPGAs. As clarified in the previous section, while the implementation was tested on a Kintex 7 FPGA, the original design resource consumption was carried out on a Virtex 7 xc7vx690. The design and every operator were optimized for performance rather than lower consumption and its resource usage can be seen in table 4.2.

Of the available resources, the most consumed were DSP slices (20 %). This is due to the fact that most of the arithmetic cores were configured for full DSP slice usage. The block memory use was minimal since it was mainly consumed by the single exponential operation. It is important to note that, given a complete use of the DSP slices available, logic slices can also be used for floating-point operations and are configured within the IP-Core. This means that a parallel implementation of more channels is feasible, considering that several FPGA's (i.e. the Ultrascale family) could posses sufficient resources for such an approach.

Finally, while the majority of acceleration results have been assessed within this chapter, a discussion on other possibilities of acceleration will be done in a global context in the conclusions section.

# 5. Conclusions

## 5.1. Hardware Acceleration of the Dispersion Interferometer Analysis

### 5.1.1. General observations and conclusions

The initial stage of this project had the goal of studying the feasibility of accelerating a basic approach to Bayesian analysis within the field of plasma diagnostics. As a prerequisite, an understanding of the difficulties and possibilities of accelerating a post-processing analysis had to be obtained. For this purpose, an example was chosen that kept some sort of simplicity while staying within the difficulty range of modern physics problems.

The dispersion interferometer (DI) was selected for this purpose given that in its model, the number of poorly known parameters as well as the parameter of interest present a low dimensionality scenario. However, the complex analytic expression for its forward model keeps the difficulty at the desired level because of its multimodality. The two possible model solutions meant that its posterior distribution presented a multimodal distribution when using a single sample.

For the problems with a multimodal posterior an online approach to Bayesian analysis, the sequential analysis, proved to be a useful tool with two main advantages for acceleration. The first is the nature of the sequential approach when compared to a batch approach, where each sample can update the posterior without the need of the full set of samples to be available. This means that the time between samples can be spent on pre-calculating other required values of the forward model. This is specifically useful for cases where signals that carry information of the forward model are constantly available, as in the case of the modulation signal coming from the PEM. The availability of this signal allowed for the pre-processing of the forward model. With this value calculated, the posterior could be immediately updated upon arrival of the following signal, thereby removing the forward model processing time from the global time.

In the case of the DI model, the rate of change of the non-free parameters in the model was on a much slower time scale than the parameter of interest. This allowed for the treatment of these parameters to be that of fixed parameters and thus permitted the dimensionality to stay one dimensional. While this seems more of a specific property of the DI, it shows that when enough knowledge of the parameters is available, some simplifications can be made to reduce the complexity of the problem. Still, this introduced an error in the estimation that required proper handling of error propagation. It was shown that the introduction of an error in a known parameter had a negligible effect on the estimation of the final density. This opens a

possibility of analyzing the effects of a parameter on the posterior to reach a decision of trading a loss in precision for a gain in acceleration.

The second advantage is in the case of inverse problems with multimodal distributions due to its forward model. When using a sequential approach, the addition of information from upcoming samples can modify the posterior distribution to solve the multimodality. This can be advantageous for problems where the rate of change of the nuisance parameter is slower than the parameter of interest. It is mainly achievable when the time to collect the necessary samples is lower than the desired time resolution of the analysis.

### 5.1.2. Acceleration of the analysis

Regarding the acceleration achieved by implementing it on FPGA architecture, besides the general advantages that a dedicated hardware implementation can have, the DI project provided insight on the feasibility of a general acceleration.

With the mentioned advantage of sequential analysis and its pre-processing, a dedicated continuous module of the forward model proves to be advantageous. It allows for a constant estimation of the forward model so that it is available to update the likelihood with the upcoming sample.

Furthermore, the high parallelization possibilities on an FPGA design provided an initial overview of the reach of this property when applied to Bayesian analysis. For the specific cases of low dimensionality, parallel modules of the same forward model can be implemented for a scan of the parameters of interest. While in the DI this allowed for a full parallelization of the single parameter scan, a multiplexing approach can be used for more resource consuming models with low dimensionality. This multiplexing, as seen in the case of the DI, comes at the cost of limiting the critical path clock frequency.

The attempt to keep a generic approach for the implementation of the forward model introduced the use of the LUT to implement complex function evaluations. This proved to be a resource intensive approach for the required precision which resulted in the limitation of the main clock frequency. Nevertheless, this limitation may be compensated by using other function evaluation methods such as CORDIC or by improving the multiplexing scheme.

For this example, the resource consumption was intentionally kept low by using fixed-point arithmetic and a limited resolution to satisfy the minimum requirements. While this was sufficient to fit the full design in a Virtex-6 FPGA that ranks within the small sized range of its family, the effects of using this arithmetic precision are difficult to judge without a comparison to another precision standard.

Based purely on the achieved speed-up, it was possible to conclude that an acceleration in the case of low dimensionality problems is feasible. The comparison with a CPU implementation showed that even at a very low initial design clock frequency, a 16-fold acceleration was achieved in the processing time of the full analysis.

Though it was not the intention of this first project to carry out a final comparison of an FPGA vs CPU, it is possible to notice some key factors. While it is reasonable to think that a 16-fold acceleration can be achieved by a CPU code running several threads, the CPU approach has two limitations: The level of parallelism achieved by the FPGA will still dominate over the CPU capabilities and while the CPU offers a significantly higher clock frequencies, it still lacks in the simultaneous modularized calculations that facilitate the pre-processing.

While both implementations could be optimized, the comparison and feasibility study were the main goal of the example and thus no deep optimization in any of them was required.

The proof of principle implementation and feasibility study of an acceleration provided enough positive results to motivate the acceleration attempt for a problem that is at par in complexity with modern physics inverse problems. The joint analysis, including the dispersion interferometer and Thomson scattering diagnostic was selected for this purpose.

## 5.2. Hardware Acceleration of the Temperature and Density Profile Analysis

Once the acceleration feasibility was demonstrated on a simpler model, the search for a Bayesian analysis of a more complex inverse problem that represents a typical example was the next step. The diversity of possible inverse problems within the field of plasma physics makes it hard to find such a representative case. For this reason, two key characteristics of Bayesian analysis of physics inverse problems were selected to represent several typical cases. One is the high number of free parameters that entails a higher dimensionality. This not only means an increase in complexity but also requires slow iterative algorithms to do the inversion. If combined with a complex forward model, each iteration makes the analysis a slow one. The other is the important advantage of Bayesian analysis to perform a joint analysis of two different physics models that share a common parameter.

The ubiquity and relevance these two factors in modern research inverse problems, helped test acceleration in a scenario representative of complex inverse problems. It was crucial to prove that one of the most useful properties of the Bayesian method, the joint analysis, is transferable to an accelerated architecture.

The joint analysis of the Dispersion interferometer and Thomson scattering presented a good test case to study these objectives. Thomson provides the complexity, dimensionality and inversion requirements to be general enough while interferometry with its simpler model, allows for the joint analysis to be tested.

### 5.2.1. Model design and software analysis

While enough data and infrastructure were available in order to use both diagnostics as a test case for this general acceleration, the initial software analysis revealed that some changes were required to be able to use these models.

The lack of raw interferometry data required the omission of its forward model from the analysis and using its processed data instead. While this seems to be a disadvantage towards the goal of certifying that the joint analysis can be accelerated, a thorough review of the implications can clarify this issue.

The omission of the DI's forward model mainly affects the resource consumption and not the total processing time. Due to the size of its forward model compared to the TS model, timing would not be affected. The operations required for the DI's model are completed significantly faster. The same goes for resource consumption when we consider the number of arithmetic operations required. The full DI forward model requires significantly less than a single channel of the TS model. Furthermore, the validation of the DI model and its analysis was already proven in the first stage. This means that the architecture can be redesigned to a higher double-precision floating-point standard with the same result expected. Finally, the previously obtained DI uncertainty can be accounted for through the uncertainty of the pre-processed line integrated electron density.

Another difficulty found during the software analysis was the case where both independent density values had significant differences. The case of bad agreement analyzed in section 4.4.3 showed that for some plasma discharges, where the density is low enough to scatter very small amounts of light, the detectors for the outer most channels had no signal but a strong background noise. This ended in a profile that exhibited high temperatures in the two channels that should measure the lower temperatures at the edge of the plasma. As explained in that section, the main reason for this was the low information content of the measured data, which specially affects the broad high temperature spectral filter. This broad bandwidth measures only background radiation and results in a data point with low signal-to-noise ratio. This high uncertainty translates in the Bayesian analysis to a broader marginal posterior for these parameters. Here the MCMC is free to explore the distribution without converging in a high

probability region within the predefined iterations. A high noise in the line integrated data from the DI meant that the addition of this information was not enough to constrain the profile to prevent this broad marginal posterior.

This problem can be addressed through two approaches. If the signal being processed were to be used for machine safety or control, the first trivial approach is to program a detection module in the FPGA with the task of recognizing the scenarios where the uncertainty is significantly higher than the mean value. A flag can be raised indicating that the data can produce profiles that can have errors. The second approach, which avoids faulty profiles, would be to constraint the priors for the problematic channels. By knowing that the plausible temperatures at the edge of the plasma, the prior can be modified to truncate or reduce the probability of implausible values. It is nevertheless important to notice that this solution comes at a risk. As mentioned in the introduction, the reason that this general Bayesian analysis is preferred to other available fast approaches is the lack of approximations and linearizations. When studying physics phenomena that are not well-known, the assumption of expecting a lower temperature in the edge could mask new information or less likely behaviors of that parameter. For example, a change in the magnetic axis could place the outermost channels in a higher temperature section of the plasma that could be unnoticed due to an over-constrained prior. A truncation or modification of the prior must be done with this factor in mind.

In general, the full software analysis proved to satisfy the selected requirements in order to attempt an acceleration with FPGA hardware architecture.

#### Hardware acceleration of an complex inverse problem

Next, a decision on which aspects of this complex problem to accelerate was made. Accelerations of inversion algorithms like Hooke and Jeeves or Markov Chain Monte Carlo are beyond the scope of this project and existing examples are discussed herein. The profiling in Minerva of a representative example like the TS model, provided insight on the weight of each section of the analysis on the total processing time. The application of Bayes' theorem is trivial enough in this design that it left little room for improvement. A reduction of iterations would be beneficial and is already achieved by modern implementations of algorithms like PT-MCMC. The acceleration of the calculations of the inversion algorithm is not crucial given that it does not contribute much processing time when compared to the other analysis elements. Therefore, the forward model becomes the main candidate and proved to be the most effective way to accelerate the global analysis for two reasons. The profiling of the Minerva code revealed that the majority of the processing time consumed was in the forward model. Since high dimensionality problems are

most popularly inverted by sampler algorithms like MCMC, each gain achieved in the forward model resulted in a gain per iteration required, vastly decreasing the global processing time.

Once selected, the acceleration of the forward model posed the question of how feasible is it to match the arithmetic precision of a CPU with FPGA architecture. Specifically, while achieving the acceleration and staying within resource usage. While this might have previously presented a problem, this project showed how current FPGA resource availability and performance allow for an implementation and acceleration of such models. Furthermore, the improvement of IP-Cores for this arithmetic simplifies the design process of an analysis that is already challenging in many other aspects.

For extensive models with complex arithmetic functions and double precision floating-point arithmetic, the FPGA was shown to provide many accelerations options which were discussed in section 1.3.2. While for some operations FPGAs can outperform CPUs, this aspect was not tested for the current analysis. The FPGA acceleration of this forward model demonstrated how complex forward models can thrive on the FPGAs parallelism and the pipelining of a hardware dedicated architecture.

In order to fully exploit these possibilities a software tool was created to decrease the time and difficulty of optimizing models with numerous complex mathematical operations. The Python code enabled this by optimizing the tracking of dependencies and simplifying the visualization of a global design for arithmetic calculations. It also provided a way to control and optimize the pipelining by taking into consideration the clock cycles required per operation and thus the buffering of factors. With these accelerations implemented in the design, a comparison with the Minerva code in Java was carried out.

**Benchmarking**

The run time differences between a Java and C implementation would normally be minuscule. The DI section already showed an acceleration higher than the expected gain achievable by implementing this full analysis in C. An implementation in C for this project would therefore not only be time-consuming but redundant when already available in Minerva. The comparison was therefore made against the Java code, ignoring special methods required by the Minerva framework and profiling only on the forward model processing time. It showed that the acceleration for a single wavelength calculation was 82-fold on an initially low critical path clock frequency. A reasonable increase of this frequency could improve the acceleration factor to 246-fold.

One has to take into account that the full analysis requires a forward model calculation for

each channel of the Thomson model. CPU multithreading can be outperformed by the higher level of parallelism that modern FPGAs allow. Specially in the Thomson model, the need for 10 spatial channels and 5 spectral increases this acceleration possibility.

If the full analysis time is approximated with the implemented clock frequency, the resulting time required for 40000 iterations of the forward model is $\sim$1.55 s for a pair of parameters in a single channel. A full parallelization of all the channels, and the fact that the forward model consumes most of the processing time, means that this estimation is close to the complete processing time of a full analysis. There are still tasks in the analysis that need to be performed, like calculating a mean and standard deviation of the chain of each free parameter. Nevertheless, most of these tasks can be updated online as soon as the representative samples are taken. Thus, their processing time would still be several orders of magnitude lower and fall within the approximated final time.

A final point of discussion would be the resource consumption. Double precision floating-point arithmetic usually results in a resource intensive architecture. This architecture was chosen to represent cases that require it. An error analysis like the one carried out in the first part of this project, could show how the use of single precision or fixed point arithmetic can suffice. The worked showed that the implementation of an extensive mathematical model allowed for the resource allocation in an FPGA which is sizable within its family but not the biggest available on the market. Based on the observed consumption, other FPGA families like the Ultrascale group can comfortably handle such a model. This type of analysis also does not require large amounts of data transfer, which hints that a multiple FPGA solution is also a possibility.

Finally, the acceleration possibilities in the other main component, the inversion algorithms, must be taken into account. In this specific analysis two common algorithms for optimization and inversion were used. As mentioned in section 2.3, accelerations of both tools are currently available. The FPGA implementation or Parallel Tempered MCMC is especially promising for this work's goal. Its multiple tempered chains could reduce the number of iterations required to reach an area of high probability, improving the other most time-consuming factor in the analysis.

## 5.3. General Aspects of the Acceleration of Bayesian Analysis

The main goal of this project was to attempt a general acceleration for this type of standard Bayesian analysis of complex non-linear inverse problems present in modern scientific experi-

ments. Specifically, those where the dynamic behavior of the parameters of interest are poorly known and therefore require the least amount of approximations and linearizations.

Finding a general solution to a problem with a large range of possible variations is an ambitious task. This meant that in order to approach a majority of the cases, available problems that shared the most characteristics with the rest were selected.

The initial approach successfully covered the cases with a simpler scenario. An on-line sequential approach was beneficial where the rate of change of the parameter of interest was lower than the data rate. While it had a smaller forward model with a low dimensionality, the model represented as well models with a non-linearity that resulted in a multimodal posterior. It showed the reach of parallelism and sequential Bayesian analysis when accelerating these type of problems, promoting further investigation of more complex models.

The second part of the project covered a bigger range of problems by targeting three common characteristics. First, an extensive forward model with numerous operations is better at representing modern inverse problems which are tackled with Bayesian analysis. Second, a multidimensional posterior which requires an iterative algorithm for the inversion of a single data set in time. Finally, it also included one of the strongest advantages of this type of analysis, which is dealing with a joint analysis of several models. Achieving this acceleration on a complex forward model with a high precision arithmetic opens the possibility of bringing this type of analyses to real-time scenarios. If mixed with modern accelerations of inversion algorithms, it is clear that while not yet in a microsecond timescale, it is closer to a fast online scenario. This is especially noticeable when comparing against the typical processing time which is generally ranging from minutes to hours.

The acceleration possibilities for several inverse problems were studied and proved achievable. This was initially carried out through the development of a tool to improve the exploitation of FPGA parallelism and pipelining. It was followed by the design and testing of a Bayesian analysis for two plasma physics diagnostics and the implementation of both analysis on FPGA hardware.

The work sets a precedent for bringing this type of analysis to hardware architectures like the FPGA. It shows that while this might not have been deemed practical a couple of years ago, modern FPGAs allow for this type of analysis. The same applies for the joint analysis of several diagnostics which was proved to be possible even if the increase in complexity is considered.

In conclusion, Bayesian analysis is a robust and powerful tool that allows for the proper estimation of parameters with a rigorous handling of their uncertainties. The increasing frequency of use in many fields of data analysis, represents its importance in the field. Accelerations have

been achieved for many specific problems, but not yet developed for the niche of this project.

Modern scientific experiments often require an unbiased inference tool that is capable of dealing with complex non-linear models where the dynamic behavior of the parameters of interest is unknown. Moreover, they often require this analysis in a faster timescale for machine control or safety. It is the intention of this work to set a precedent towards bringing a general approach of this analysis closer to a real-time frame and to promote the study of more acceleration possibilities.

# A. Appendix

## A.1. Hardware Optimization Tool Workflow

The work-flow for the use of the developed Python tool is the following:

Once the factorization of the model's equation has been selected, it is arranged to separate constants from variables and, in the case of typical hardware implementations, reduce the number of divisions necessary. Equation (A.1) shows how the Bessel function is modified for this purpose.

$$\begin{aligned} f(2\alpha) &\approx \sqrt{\frac{\pi}{2(2\alpha)}}\left(1+\frac{15}{8(2\alpha)}+\frac{105}{128(2\alpha)^2}+\frac{315}{1024(2\alpha)^3}\right) \\ f(2\alpha) &\approx \sqrt{\frac{K_1}{(2\alpha)}}\left(1+\frac{K_2}{(2\alpha)}+\frac{K_3}{(2\alpha)^2}+\frac{K_4}{(2\alpha)^3}\right) \\ \frac{1}{2\alpha} &= T_e r_e E_{ei} \end{aligned} \tag{A.1}$$

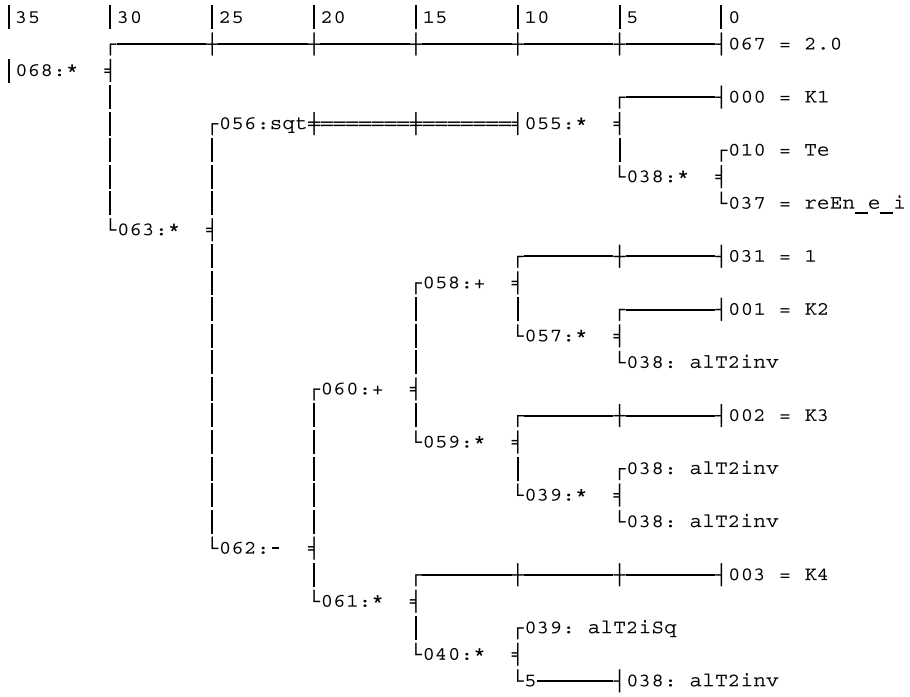
With the expression selected, the equation can be entered following the syntax described in section 2.4 as shown in eq. (A.2). In this case, the equation was not given in a single line but rather a breakdown of it. With this approach the analysis is simplified by giving parent nodes specific names that appear when the value is used, like *alphaT2inv* in fig. A.1.

$$\begin{aligned} &\text{R} && = [['\text{K1}', 0], ['\text{K2}', 0], ['\text{K3}', 0], ['\text{K4}', 0] \\ &\text{alphaT2inv} && = \text{Solver}(\text{Te} * \text{reEn\_e\_i}, \text{R}) \\ &\text{alphaT2iSq} && = (\text{alphaT2inv} * \text{alphaT2inv}) \\ &\text{alphaT2iCub} && = (\text{alphaT2iSq} * \text{alphaT2inv}) \\ &\text{bessel1} && = ('\text{K1}' * \text{alphaT2inv}).\text{fun}('\text{sqrt}', '\text{bessel1}') \\ &\text{bessel2} && = ('1.0' +' \text{K2}' * \text{alphaT2inv} +' \text{K3}' * (\text{alphaT2iSq}) -' \text{K4}' * (\text{alphaT2iCub})) \\ &\text{bessel} && = \text{bessel1} * \text{bessel2} \\ &\text{factor2} && =' 2.0' * \text{bessel} \end{aligned} \tag{A.2}$$

In the previous code the first line represents the input of alphanumeric constants or any value that will represent a leaf in the graph. The comma separated number represents the delay the signal will have.

The first input of the arithmetic expression uses the $Solver()$ method to initially pass the library with the predefined constants. The $.fun()$ method is also used to represent special functions like the square root in this case which is not recognized by the parser.

The output of this code example is shown in fig. A.1 where a ASCII character represents the tree representing the dependencies, latencies and critical path of the implemented model.




```
0: 004, 005, 006, 007, 008, 009, 011, 012, 013, 014, 015, 016, 017, 018, 019,
020, 021, 022, 023, 024, 025, 026, 027, 028, 029, 030, 032, 033, 034, 035,
036, 041, 042, 043, 044, 045, 046, 047, 048, 049, 050, 051, 052, 053, 054,
064, 065, 066
1: 000, 001, 002, 003, 010, 031, 037, 040, 055, 056, 057, 058, 059, 060, 061,
062, 063, 067, 068
2: 039
5: 038
```

Figure A.1.: Tool example output of the Bessel function implementation.

## A.2. Hardware function refactorization

This section contains the relevant functions that were arithmetically manipulated to suit better the FPGA hardware implementation.

The $1/(\epsilon + 1)$ factor which is a function of the scattered wavelength was generated by expressing it the same way that the wavelength changed can be expressed. That is an initial wavelength value plus a wavelength increment until the range is complete

$$\lambda_{s_i} = \lambda_{s_0} + i\Delta\lambda_s \tag{A.3}$$

$$\frac{1}{\epsilon_i + 1} = \frac{\lambda_l}{\lambda_{s_0}} + i\frac{\Delta\lambda_s\lambda_l}{(\lambda_{s_0} + \Delta\lambda_s)\lambda_{s_0}} \tag{A.4}$$

The other equations in the Naito model, modified to promote parallelism and reduce resource consumption were:

$$S_z(\epsilon, \theta, \alpha) = \frac{1}{K_2^*}\left(\frac{1}{1+\epsilon}\right)^3 \frac{1}{2\sqrt{2[1-\cos(\theta)](1+\epsilon)+\epsilon^2}} e^{(2\alpha)(1-x)} \tag{A.5}$$

$$K_2^*(2\alpha) \approx \sqrt{K_1\frac{1}{(2\alpha)}}\left[1 + K_2\left(\frac{1}{2\alpha}\right) + K_3\left(\frac{1}{2\alpha}\right)^2 + K_4\left(\frac{1}{2\alpha}\right)^3\right] \tag{A.6}$$

$$u = \frac{\sin\theta}{1-\cos\theta} = K_u, \qquad x = \sqrt{1 + \frac{\epsilon^2}{2[K_{cTheta}](1+\epsilon)}} \tag{A.7}$$

$$2\alpha = \left(\frac{r_e E_r}{T_e}\right), \quad \epsilon = \frac{\lambda_t - \lambda_l}{\lambda_l} \tag{A.8}$$

$$\begin{aligned}
q_0 &= p_0 = p_{0a} + p_{0b}\zeta^2 + (p_{0c})\zeta^4, \\
q_1 &= q_{1a}\zeta^3(q_{1b} + q_{1c}\zeta^2), \\
p_1 &= \zeta(p_{1a}\zeta^2 + p_{1b}\zeta^4 + p_{1c}), \\
q_2 &= (q_{2a} + q_{2b}\zeta^2 + q_{2c}\zeta^4 + q_{2d}\zeta^6), \\
p_2 &= (p_{2a} + p_{2b}\zeta^2 + p_{2c}\zeta^4 + p_{2d}\zeta^6),
\end{aligned} \tag{A.9}$$

## A.3. Temperature and Density Histograms for Profile Inference

When analyzing the chain of each parameter, the histogram of that chain provides a marginalized posterior of its parameter as shown in fig. A.2.

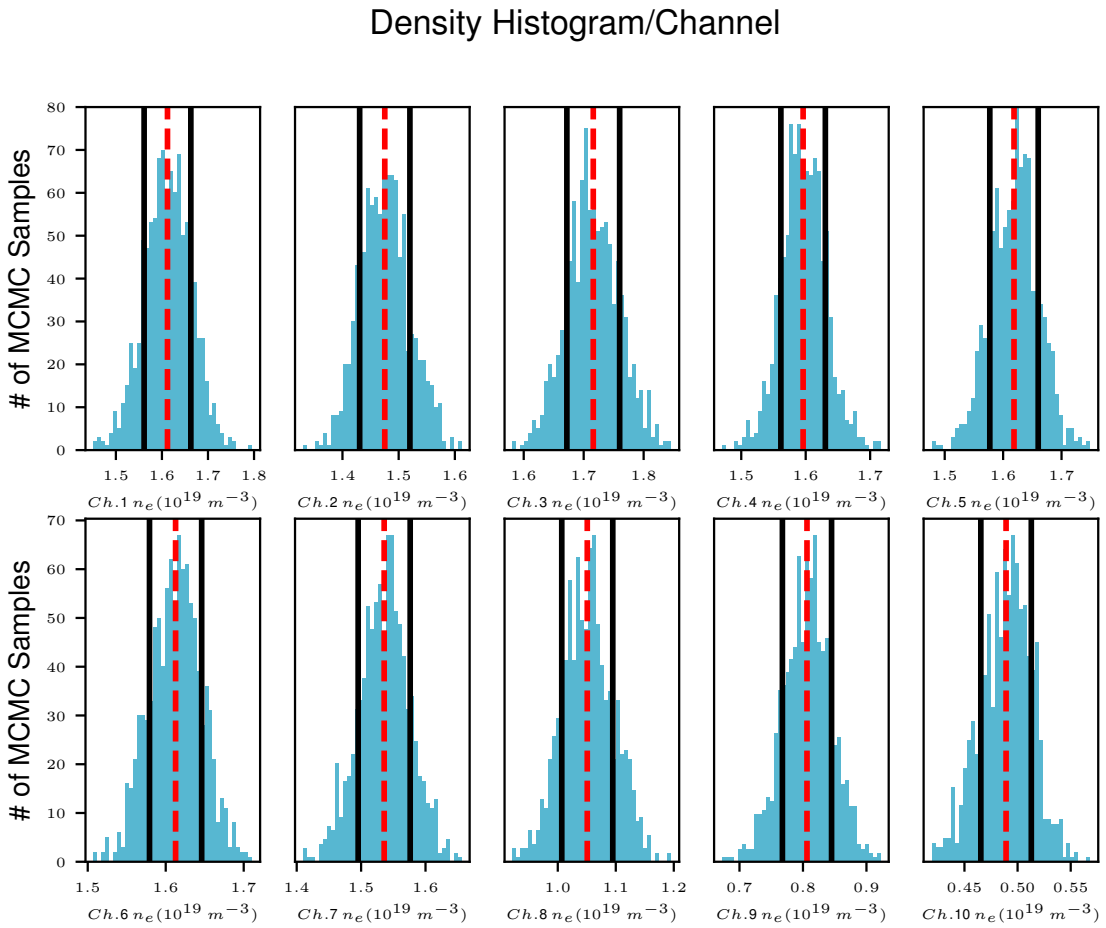



Figure A.2.: Density histograms per channel for good previous agreement with the joint analysis mean value (red) and its standard deviation (black)

This can be used to obtain a mean and standard deviation used to build the profile, as depicted in section 4.4.3, or can be plotted several ways depending on the analysis required. Another representation is shown in fig. A.3.

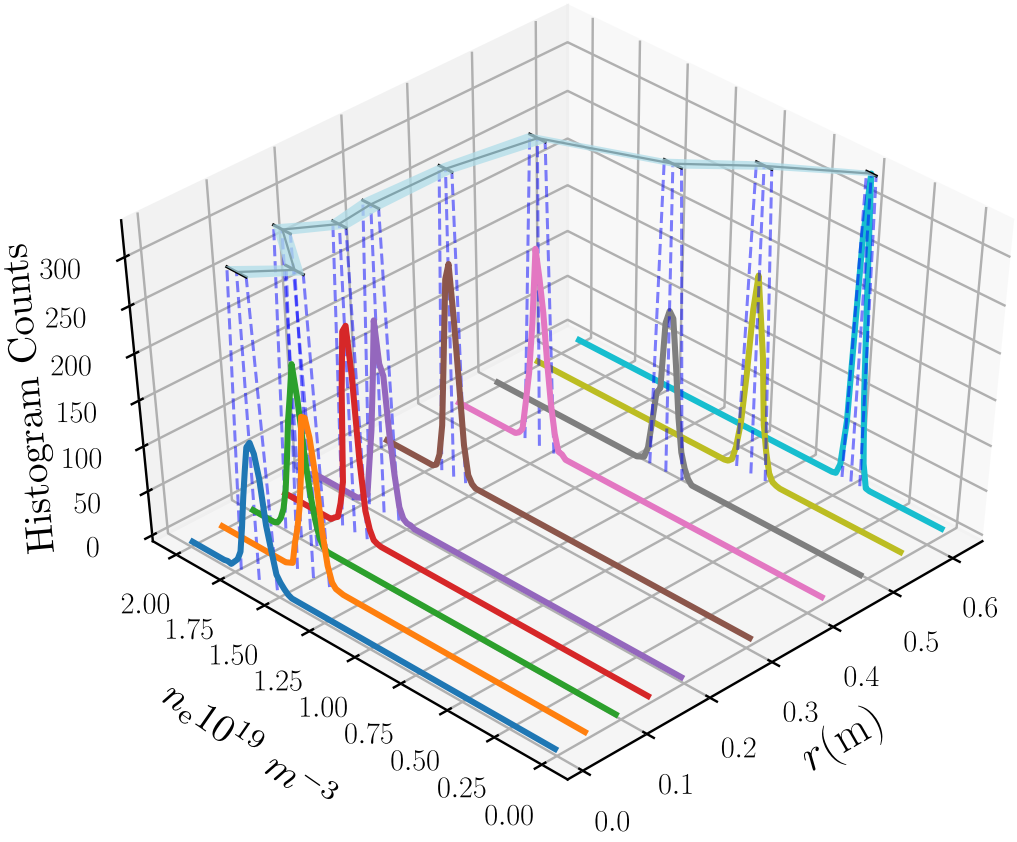



Figure A.3.: Density histogram outline per channel in a 3D view.

# B. Glossary - Terms and Acronyms

| Term/Acronym | Brief description | Scope | Page |
|---|---|---|---|
| D | Deuterium: Hydrogen isotope | Document | 19 |
| T | Tritium: Hydrogen isotope | Document | 19 |
| W7-X | Wendelstein 7-X | Document | 21 |
| TS | Thomson scattering | Diagnostics | 22 |
| PROM | Programmable Read-Only Memories | Document | 23 |
| PLD | Programmable Logic Devices | Document | 23 |
| FPGA | Field Programmable Gate Arrays | Document | 23 |
| I/O | Input/Output | Document | 23 |
| CLB | Configurable Logic Block | Document | 23 |
| SM | Switch Matrix | Document | 23 |
| IO | Input/Output | Document | 23 |
| LUT | Look-Up Table | Document | 24 |
| DSP | Digital Signal Processing | Document | 24 |
| HDL | hardware description language | Document | 24 |
| ASIC | application-specific integrated circuit | Document | 24 |
| VHSIC | very high speed integrated circuit | Document | 24 |
| VHDL | VHSIC hardware description language | Document | 24 |
| GPU | graphics processing unit | Document | 25 |
| IP-Core | Intellectual property core | Document | 25 |
| RAM | random access memory | Document | 25 |
| FM | Forward Model | Analysis | 30 |
| PDF | Probability Density Function | Analysis | 31 |
| MAP | Maximum a Posteriori | Analysis | 34 |
| LGI | Linear Gaussian Inversion | Analysis | 34 |
| Minerva | Bayesian analysis framework | Analysis | 34 |
| MCMC | Markov Chain Monte Carlo | Analysis | 36 |
| PT-MCMC | Parallel Tempering MCMC | Analysis | 37 |
| Pipelining | Hardware architecture strategy | Analysis | 39 |
| Root node | Base or superior node of a tree structure | Analysis | 41 |
| Leaf node | End node in a tree structure | Analysis | 41 |
| DI | Dispersion interferometer | Diagnostics | 46 |
| ZnSe | Zinc selenide | Diagnostics | 48 |
| $AgGaSe_2$ | Silver Selenogallate | Diagnostics | 48 |
| FDC | Frequency Doubling Crystal | Diagnostics | 48 |
| PEM | Photo-Elastic Modulator | Diagnostics | 49 |
| CCR | corner cube retroreflector | Diagnostics | 49 |
| $MgF_2$ | Magnesium fluoride | Diagnostics | 50 |
| ADC | Analog to Digital Converter | Diagnostics | 50 |
| TTL | Transistor-Transistor Logic | Implementation | 66 |
| CORDIC | Coordinate Rotation Digital Computer | Implementation | 74 |

# Bibliography


[1] H. T. Mora et al. "Acceleration of Bayesian model based data analysis". In: *2016 IEEE 13th International Conference on Signal Processing (ICSP)*. Nov. 2016, pp. 523–529. DOI: 10.1109/ICSP.2016.7877889.

[2] H. T. Mora et al. "FPGA acceleration of Bayesian model based analysis for time-independent problems". In: *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Nov. 2017, pp. 774–778. DOI: 10.1109/GlobalSIP.2017.8309065.

[3] Steven N. Evans and Philip B. Stark. "Inverse problems as statistics". In: *Inverse Problems* 18.4 (2002). ISSN: 02665611. DOI: 10.1088/0266-5611/18/4/201.

[4] Ashley H Carter. "A class of inverse problems in physics". In: *American Journal of Physics* 68.8 (2000), pp. 698–703. ISSN: 00029505. DOI: 10.1119/1.19530. URL: http://link.aip.org/link/?AJP/68/698/1.

[5] R Fischer et al. "Bayesian background estimation". In: *AIP Conference Proceedings* (2001), pp. 193–212. ISSN: 0094243X. DOI: 10.1063/1.1381857. URL: papers://8a7341e4-9daa-4c0d-bcf2-2d9b813974af/Paper/p9863.

[6] Fabrizia Guglielmetti, Rainer Fischer, and Volker Dose. "Bayesian mixture models for poisson astronomical images". In: *Information Systems Development: Reflections, Challenges and New Directions*. 2013, pp. 197–202. ISBN: 9781461449508. DOI: 10.1007/978-1-4614-3520-4-18. arXiv: 1202.0390.

[7] J. Svensson et al. "Modelling of JET Diagnostics Using Bayesian Graphical Models". In: *Contributions to Plasma Physics* 51.2-3 (2011), pp. 152–157. ISSN: 08631042. DOI: 10.1002/ctpp.201000058.

[8] J. Svensson et al. "Connecting physics models and diagnostic data using Bayesian Graphical Models". In: *37th EPS Conference on Plasma Physics* 10 (2010), O4.117.

[9] M. Lennholm et al. "ELM frequency feedback control on JET". In: *Nuclear Fusion* 55.6 (2015), p. 063004. ISSN: 0029-5515. DOI: 10.1088/0029-5515/55/6/063004. URL: http://iopscience.iop.org/article/10.1088/0029-5515/55/6/063004.

[10] V Dose. "Bayesian inference in physics: case studies". In: *Reports on Progress in Physics* 66.9 (2003), p. 1421. URL: http://stacks.iop.org/0034-4885/66/i=9/a=202.

[11] Jorge Lobo and João Filipe Ferreira. "Special Issue on Unconventional computing for Bayesian inference". In: *International Journal of Approximate Reasoning* 88 (June 2017). DOI: 10.1016/j.ijar.2017.06.004.

[12] Simo Sarkka. "Bayesian Filtering and Smoothing". In: *Cambridge University Press* (2013), p. 254. DOI: 10.1017/CBO9781139344203. URL: http://dl.acm.org/citation.cfm?id=2534502%7B%5C%%7D5Cnhttp://ebooks.cambridge.org/ref/id/CBO9781139344203.

[13] Z H E Chen. "Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond". In: *Statistics* 182.1 (2003), pp. 1–69. ISSN: 00220949. DOI: 10.1.1.107.7415. arXiv: 10.1.1.107.7415. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.7415%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf.

[14] M. Sanjeev Arulampalam et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking". In: *IEEE Transactions on Signal Processing* 50.2 (2002), pp. 174–188. ISSN: 1053587X. DOI: 10.1109/78.978374. arXiv: arXiv:1011.1669v3.

[15] M.A. Chappell et al. "Variational Bayesian Inference for a Nonlinear Forward Model". In: *IEEE Transactions on Signal Processing* 57.1 (2009), pp. 223–236. ISSN: 1053-587X. DOI: 10.1109/TSP.2008.2005752.

[16] Simo Sarkka and Aapo Nummenmaa. "Recursive noise adaptive Kalman filtering by variational Bayesian approximations". In: *IEEE Transactions on Automatic Control* 54.3 (2009), pp. 596–600. ISSN: 00189286. DOI: 10.1109/TAC.2008.2008348.

[17] Y. Ho and R. Lee. "A Bayesian approach to problems in stochastic estimation and control". In: *IEEE Transactions on Automatic Control* 9.4 (1964), pp. 333–339. ISSN: 0018-9286. DOI: 10.1109/TAC.1964.1105763. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1105763.

[18] Lifeng Miao et al. "Efficient bayesian tracking of multiple sources of neural activity: Algorithms and real-time FPGA implementation". In: *IEEE Transactions on Signal Processing* 61.3 (2013), pp. 633–647. ISSN: 1053587X. DOI: 10.1109/TSP.2012.2226172.

[19] Lyman Spitzer. "The Stellarator Concept". In: *IEEE Transactions on Plasma Science* 9.4 (1981), pp. 130–141. ISSN: 19399375. DOI: 10.1109/TPS.1981.4317418.

[20] F F Chen. *Introduction to Plasma Physics and Controlled Fusion*. Introduction to Plasma Physics and Controlled Fusion v. 1. Springer, 1984. ISBN: 9780306413322. URL: https://books.google.de/books?id=ToAtqnznr80C.

[21] S M Casselman. *FPGA virtual computer for executing a sequence of program instructions by successively reconfiguring a group of FPGA in response to those instructions*. 1997. URL: https://www.google.com/patents/US5684980.

[22] Wim and Roelandts. "Putting Things in Perspective". In: *THE Quarterly Journal for Programmable Logic Users XCELL* 32 (1999), p. 4. ISSN: 0029-2915 (Print). DOI: 10.1126/science.326.5955.919-c. URL: https://www.xilinx.com/publications/archives/xcell/Xcell32.pdf.

[23] User Guide. "7 Series FPGAs Configurable Logic Block". In: 474 (2016).

[24] Xilinx Inc. "7 Series DSP48E1 Slice". In: *Design* 479 (2012), pp. 1–56.

[25] Douglas J. Smith. *HDL Chip Design: A Practical Guide for Designing, Synthesizing and Simulating ASICs and FPGAs Using VHDL or Verilog*. Doone Publications, 1998. ISBN: 0965193438.

[26] M. C. Herbordt et al. "Achieving High Performance with FPGA-Based Computing". In: *Computer* 40.3 (Mar. 2007), pp. 50–57. ISSN: 0018-9162. DOI: 10.1109/MC.2007.79.

[27] F. B. Muslim et al. "Efficient FPGA Implementation of OpenCL High-Performance Computing Applications via High-Level Synthesis". In: *IEEE Access* 5 (2017), pp. 2747–2762. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2017.2671881.

[28] Altera. "White Paper Designing and Using FPGAs for Double-Precision Floating-Point Math". In: *Altera* August (2007), pp. 1–6. URL: https://www.altera.com/en%7B%5C_%7DUS/pdfs/literature/wp/wp-01028.pdf.

[29] John D. Cappello and Dave Strenski. "A practical measure of FPGA floating point acceleration for High Performance Computing". In: *Proceedings of the International Conference on Application-Specific Systems, Architectures and Processors* (2013), pp. 160–167. ISSN: 10636862. DOI: 10.1109/ASAP.2013.6567570.

[30] J. Svensson et al. "Integrating diagnostic data analysis for W7-AS using Bayesian graphical models". In: *Review of Scientific Instruments* 75.10 II (2004), pp. 4219–4221. ISSN: 00346748. DOI: 10.1063/1.1789611.

[31] J. Svensson and A. Werner. "Current tomography for axisymmetric plasmas". In: *Plasma Physics and Controlled Fusion* 50.8 (2008). ISSN: 07413335. DOI: 10.1088/0741-3335/50/8/085002.

[32] J Svensson and A Werner. "Large Scale Bayesian Data Analysis for Nuclear Fusion Experiments". In: *IEEE International Symposium on Intelligent Signal Processing* 07 (2007), pp. 1–6. DOI: 10.1109/WISP.2007.4447579. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4447579.

[33] Antonio Roldao Lopes and George A. Constantinides. "A high throughput FPGA-Based floating point conjugate gradient implementation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4943 LNCS (2008), pp. 75–86. ISSN: 03029743. DOI: 10.1007/978-3-540-78610-8_10.

[34] Robert Hooke and T A Jeeves. "" Direct Search" Solution of Numerical and Statistical Problems". In: *J. ACM* 8.2 (Apr. 1961), pp. 212–229. ISSN: 0004-5411. DOI: 10.1145/321062.321069. URL: http://doi.acm.org/10.1145/321062.321069.

[35] Robert Michael Lewis, Virginia Torczon, and Michael W. Trosset. "Direct search methods: then and now". In: *Journal of Computational and Applied Mathematics* 124.1 (2000). Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations, pp. 191–207. ISSN: 0377-0427. DOI: https://doi.org/10.1016/S0377-0427(00)00423-4. URL: http://www.sciencedirect.com/science/article/pii/S0377042700004234.

[36] W K Hasting. "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1 (1970), pp. 97–109. DOI: 10.1093/biomet/57.1.97.

[37] Rh Swendsen and Js Wang. "Replica Monte Carlo simulation of spin glasses". In: *Phys. Rev. Lett.* 57.21 (1986), pp. 2607–2609. ISSN: 1079-7114. DOI: 10.1103/PhysRevLett.57.2607. URL: http://www.ncbi.nlm.nih.gov/pubmed/10033814.

[38] Grigorios Mingas and Christos Savvas Bouganis. "A custom precision based architecture for accelerating parallel tempering MCMC on FPGAs without introducing sampling error". In: *Proceedings of the 2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines, FCCM 2012* (2012), pp. 153–156. DOI: 10.1109/FCCM.2012.34.

[39] Grigorios Mingas and Christos Savvas Bouganis. "Parallel tempering MCMC acceleration using reconfigurable hardware". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7199 LNCS (2012), pp. 227–238. ISSN: 03029743. DOI: 10.1007/978-3-642-28365-9_19.

[40] Grigorios Mingas, Farhan Rahman, and Christos Savvas Bouganis. "On optimizing the arithmetic precision of MCMC algorithms". In: *Proceedings - 21st Annual International IEEE Symposium on Field-Programmable Custom Computing Machines, FCCM 2013* (2013), pp. 181–188. DOI: 10.1109/FCCM.2013.31.

[41] Behrooz Parhami. *Computer Architecture: From Microprocessors to Supercomputers (Oxford Series in Electrical and Computer Engineering)*. New York, NY, USA: Oxford University Press, Inc., 2005. ISBN: 019515455X.

[42] J. Knauer et al. "A New Dispersion Interferometer for the Stellarator Wendelstein 7-X". In: *43 EPS Conference on Plasma Physics* (2016).

[43] P. Kornejew et al. "Final Design of the Dispersion Interferometer for the Wendelstein7-X Stellarator". In: *40th EPS Conference on Plasma Physics* figure 1 (2012), pp. 10–13.

[44] A. J.H. Donné. "High spatial resolution interferometry and polarimetry in hot plasmas". In: *Review of Scientific Instruments* 66.6 (1995), pp. 3407–3423. ISSN: 00346748. DOI: 10.1063/1.1145516.

[45] T. Akiyama et al. "Development of CO2 laser dispersion interferometer with photoelastic modulator". In: *Review of Scientific Instruments* 81.10 (2010). ISSN: 00346748. DOI: 10.1063/1.3460453.

[46] P. A. Bagryansky et al. "Dispersion interferometer based on a C O2 laser for TEXTOR and burning plasma experiments". In: *Review of Scientific Instruments* 77.5 (2006). ISSN: 00346748. DOI: 10.1063/1.2202922.

[47] V. P. Drachev, Yu I. Krasnikov, and P. A. Bagryansky. "Dispersion interferometer for controlled fusion devices". In: *Review of Scientific Instruments* 64.4 (1993), pp. 1010–1013. ISSN: 00346748. DOI: 10.1063/1.1144170.

[48] A. Lizunov et al. “Development of a multichannel dispersion interferometer at TEXTOR”. In: *Review of Scientific Instruments*. Vol. 79. 10. 2008. ISBN: 1089-7623 (Electronic)\r0034-6748 (Linking). DOI: 10.1063/1.2969466.

[49] Stange, Torsten et al. “Advanced electron cyclotron heating and current drive experiments on the stellarator Wendelstein 7-X”. In: *EPJ Web Conf.* 157 (2017), p. 2008. DOI: 10.1051/epjconf/201715702008. URL: https://doi.org/10.1051/epjconf/201715702008.

[50] D Veron. “Submillimeter interferometry of high-density plasmas”. In: *Infrared and millimeter waves. Volume 2. (A80-34968 14-35) New York, Academic Press, Inc., 1979, p. 67-135.* Vol. 2. 1979, pp. 67–135.

[51] T Akiyama et al. “Conceptual Design of dispersion interferometer using ratio of modulation amplitudes P2-34”. In: *Proceedings of ITC18,2008*. 2008, pp. 401–405. DOI: 10.1585/pfr.5.S1041.

[52] Devinderjit Sivia and John Skilling. “Data Analysis: A Bayesian Tutorial”. In: *Technometrics* 40.2 (1998), p. 155. ISSN: 00401706. DOI: 10.2307/1270652. URL: http://www.amazon.com/dp/0198568320%7B%5C%%7D5Cnhttp://www.jstor.org/stable/1270652?origin=crossref.

[53] Iosifina Pournara, Christos S. Bouganis, and George A. Constantinides. “FPGA-accelerated Bayesian learning for reconstruction of gene regulatory networks”. In: *Proceedings - 2005 International Conference on Field Programmable Logic and Applications, FPL* 2005 (2005), pp. 323–328. DOI: 10.1109/FPL.2005.1515742.

[54] Y. H. Hu. “CORDIC-based VLSI architectures for digital signal processing”. In: *IEEE Signal Processing Magazine* 9.3 (July 1992), pp. 16–35. ISSN: 1053-5888. DOI: 10.1109/79.143467.

[55] Dirk Timmermann, Helmut Hahn, and Bedrich J. Hosticka. “Low Latency Time CORDIC Algorithms”. In: *IEEE Transactions on Computers* 41.8 (1992), pp. 1010–1015. ISSN: 00189340. DOI: 10.1109/12.156543.

[56] E. Pasch et al. “The Thomson scattering system at Wendelstein 7-X”. In: *Review of Scientific Instruments* 87.11 (2016). ISSN: 10897623. DOI: 10.1063/1.4962248.

[57] I. H. Hutchinson. *Principles of Plasma Diagnostics*. 2005. DOI: 10.1017/CBO9780511613630. arXiv: arXiv:1011.1669v3. URL: http://books.google.com.hk/books?id=pUUZKLR0ORIC.

[58] O. Naito, H. Yoshida, and T. Matoba. “Analytic formula for fully relativistic Thomson scattering spectrum”. In: *Physics of Fluids B: Plasma Physics* 5.11 (1993), pp. 4256–4258. ISSN: 0899-8221. DOI: 10.1063/1.860593. URL: http://aip.scitation.org/doi/10.1063/1.860593.

[59] S. A. Bozhenkov et al. “The Thomson scattering diagnostic at Wendelstein 7-X and its performance in the first operation phase”. In: *Journal of Instrumentation* 12.10 (2017). ISSN: 17480221. DOI: 10.1088/1748-0221/12/10/P10004.

[60] Mary Kathryn Cowles and Bradley P. Carlin. “Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review”. In: *Journal of the American Statistical Association* 91.434 (1996), pp. 883–904. ISSN: 01621459. URL: http://www.jstor.org/stable/2291683.

[61] G. O. Roberts, A. Gelman, and W. R. Gilks. “Weak convergence and optimal scaling of random walk Metropolis algorithms”. In: *Ann. Appl. Probab.* 7.1 (Feb. 1997), pp. 110–120. DOI: 10.1214/aoap/1034625254. URL: https://doi.org/10.1214/aoap/1034625254.

[62] Jeffrey S. Rosenthal. “Optimal Proposal Distributions and Adaptive MCMC”. In: 2008.