UNIVERSIDAD POLITÉCNICA DE MADRID

E.T.S. DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

PROYECTO FIN DE MÁSTER

MÁSTER UNIVERSITARIO EN APRENDIZAJE AUTOMÁTICO Y DATOS MASIVOS

Improving Convergence Detection for Plasma Transport Simulations through Evolutionary Optimization

Desarrollado por: Adrián Martín Sanabria

Dirigido por: Ángel Panizo Lledot y Cristian Ramírez Atencia

Madrid, November 8, 2025

Improving Convergence Detection for Plasma Transport Simulations through Evolutionary Optimization
Desarrollado por: Adrián Martín Sanabria
Dirigido por: Ángel Panizo Lledot y Cristian Ramírez Atencia
Proyecto Fin de Máster, November 8, 2025

E.T.S. de Ingeniería de Sistemas Informáticos
Campus Sur UPM, Carretera de Valencia (A-3), km. 7
28031, Madrid, España

Si deseas citar este trabajo, la entrada completa en BibTeX es la siguiente:

```
@mastersthesis{citekey,
    title = {Improving Convergence Detection for Plasma Transport Simulations
through Evolutionary Optimization},
    author = {Martín Sanabria, A.},
    school = {E.T.S. de Ingeniería de Sistemas Informáticos},
    year = {2025},
    month = {11},
}
```

Acknowledgements

I am especially thankful to Pablo Rodríguez Fernández for giving me the opportunity to work on such an amazing project, for his understanding and of course for the crash course in fusion and plasma science. I would also like to thank my roommate, lab-mate and friend Javier whose help in this project physics-wise was definitely necessary. A special thanks to my parents, whose support helped me through this entire project. And last and definitely not least, thanks to Ale who listened to my "yapping" and gave me the strength I needed to keep going.

Resumen

La fusión nuclear es una de las alternativas más prometedoras para alcanzar una fuente de energía limpia, segura y prácticamente inagotable. Sin embargo, la complejidad y el elevado coste computacional de las simulaciones dificultan la predicción del comportamiento del plasma dentro de los reactores de fusión, lo que ralentiza enormemente el avance de esta tecnología. El presente trabajo aborda la calibración de la métrica Ricci que es utilizada en PORTALS, un framework desarrollado por el PSFC del MIT que combina modelos subrogados basados en procesos gaussianos con optimización bayesiana para acelerar las simulaciones de transporte de plasma.

El objetivo principal del proyecto ha sido optimizar los parámetros d_0 y λ de dicha métrica para mejorar la detección automática de la convergencia en las simulaciones de PORTALS. Para ello, se aplicó un algoritmo genético configurado con validación cruzada en 5 folds y 32 ejecuciones por fold, evaluando la estabilidad de los parámetros y la robustez de la calibración.

Los resultados demuestran que la configuración optimizada $(d_0 = 1.97, \lambda = 0.17)$ reduce el error de detección de convergencia en un 74% respecto a la métrica base y un 44% frente a la versión calibrada manualmente, sin introducir falsos positivos.

Se concluye que la métrica calibrada por el algoritmo genético mejora la fiabilidad y reproducibilidad del criterio de parada, y acelera la detección de convergencia respecto a los parámetros por defecto.

Palabras clave: fusión nuclear; simulación de plasma; modelos subrogados; optimización genética; métrica de Ricci; PORTALS

Abstract

Nuclear fusion is one of the most promising alternatives for achieving a clean, safe, and virtually inexhaustible energy source. However, the complexity and high computational cost of simulations make it difficult to predict plasma behavior inside fusion reactors, which greatly slows down the progress of this technology. This work addresses the calibration of the Ricci metric used in PORTALS, a framework developed by the PSFC at MIT that combines surrogate models based on Gaussian Processes with Bayesian Optimization to accelerate plasma transport simulations.

The main objective of the project was to optimize the parameters d_0 and λ of this metric to improve the automatic detection of convergence in PORTALS simulations. To this end, a genetic algorithm was applied, configured with 5-fold cross-validation and 32 runs per fold, assessing the stability of the parameters and the robustness of the calibration.

The results show that the optimized configuration $(d_0 = 1.97, \lambda = 0.17)$ reduces the convergence detection error by 74% compared to the baseline metric and by 44% compared to the manually calibrated version, without introducing false positives.

It is concluded that the metric calibrated by the genetic algorithm improves the reliability and reproducibility of the stopping criterion and accelerates convergence detection compared to the default parameters.

Keywords: nuclear fusion; plasma simulation; surrogate modeling; genetic optimization; Ricci metric; PORTALS

Contents

List of Figures

List of Tables

1. Introduction

The pursuit of nuclear fusion as a source of energy dates back to the mid 20th century. This technology has since been regarded as one of, if not the most, promising paths to achieve a safe, abundant and sustainable energy source. Over the years, researchers have explored a plethora of approaches to harness the power of the Sun, including inertial confinement, magneto-inertial confinement and magnetic confinement. This last approach consist of the usage of electromagnets to confine high-temperature plasma, preventing it from coming into direct contact with the walls of the reactor and thus maintaining the conditions necessary for fusion to occur.

Among magnetic confinement approaches, tokamaks and stellarators (see Figure 1.1) have emerged as the leading reactor concepts, differing mainly in that the former relies on a plasma current for confinement, while the latter achieves steady-state operation using only external magnetic coils

Modeling the behavior of these magnetic confinement systems requires advanced numerical tools capable of capturing the complex, coupled physics of plasma transport. State-of-the-art gyrokinetic multiscale simulations can demand on the order of 10^7 CPU hours per case [1], making systematic exploration and validation prohibitively expensive without the use of surrogate or reduced models.

Recent advances in artificial intelligence have shown their potential to address safety-

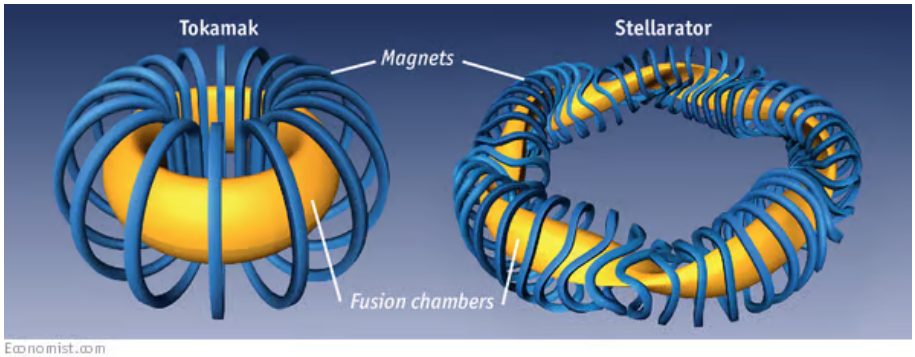

Figure 1.1. Schematic comparison between the main magnetic confinement concepts: tokamak and stellarator

critical challenges in fusion research. Deep-learning models have demonstrated cross-machine generalization in disruption forecasting across major tokamaks such as JET and DIII-D [2], illustrating how data-driven approaches can complement traditional physics-based modeling.

Beyond disruption prediction, recent community position papers have outlined high-impact opportunities for applying artificial intelligence across fusion research, including diagnostics, control, simulation, and materials discovery [3].

In the realm of predictive modeling of plasma behavior, achieving a steady-state solution in transport solvers represents a crucial achievement. The PORTALS framework [4], created by the Massachusetts Institute of Technology (MIT) Plasma Science and Fusion Center (PSFC), utilizes surrogate modeling and optimization techniques. This approach facilitates the prediction of core plasma profiles and performance using nonlinear gyrokinetic simulations at a greatly reduced computational cost, without compromising accuracy.

In this setting, a key challenge is determining when a simulation has reached a physically significant steady state, a task that becomes particularly complex in multichannel, interconnected transport situations where assessing convergence isn't simple. Conventional residual metrics can be helpful, but they often overlook the uncertainty present in surrogate-based optimization.

To tackle this challenge, PORTALS employs the χR metric (hereafter referred to as the Ricci metric) [5] as a scalar indicator of the difference between predicted and actual transport quantities, incorporating their uncertainties. Based on a probabilistic framework, the Ricci metric assesses the statistical distinguishability of two distributions. Its functionality is influenced by two adjustable hyperparameters, d_0 and λ, which determine the sensitivity and scaling of the metric. These parameters are frequently chosen heuristically, which may restrict the Ricci metric's clarity or efficacy as a convergence tool.

1.1. Objectives

This thesis seeks to refine the parameters d_0 and λ of the Ricci metric for it to serve as a dependable indicator of plasma convergence over a broad spectrum of simulation scenarios. Specifically, the aim is to guarantee that when the Ricci metric is beneath

a specified threshold, the plasma state in question not only achieves numerical convergence (as indicated by flux residuals) but also resides within a defined confidence range, such as 2 standard deviations from the steady-state solution. The objectives include:

- Establishing a physically meaningful threshold for what constitutes "converged" solutions,
- Creating a Ricci metric that applies across various plasma regimes and data collection methodologies,
- Developing a more automated and well-founded stopping criterion for surrogate-driven optimization within transport solvers.

1.2. Motivation

Fossil fuels such as oil, coal, and natural gas currently account for nearly 80% of global primary energy consumption, according to data from the Energy Institute's [6] (see Figure 1.2). These sources of energy are the main cause of global warming as they release large amounts of greenhouse gases like carbon dioxide into the atmosphere [7], [8].

As global energy demand continues to grow, having almost tripled in the last six decades, the reliance on these carbon-intensive sources has only deepened. In order to keep up with this tendency while mitigating climate change, it is essential to develop an energy source that can provide enough energy to replace fossil fuels without contributing to global warming.

Nuclear fusion is a promising candidate for achieving this type of energy, it offers the potential to generate large scale energy without releasing any greenhouse gasses or producing any long-lasting radioactive waste. Nuclear fusion is still being researched and advances in this technology depend on being able to simulate plasma behavior in certain conditions to complement reactor development. However, this type of simulations present a challenge as they are highly computationally costly.

The development of surrogate modeling frameworks such as PORTALS directly contributes to overcoming these computational challenge. The framework is designed to accelerate the search for steady-state solutions by enhancing transport solvers' predictive capabilities with machine learning models. It combines surrogate modeling with

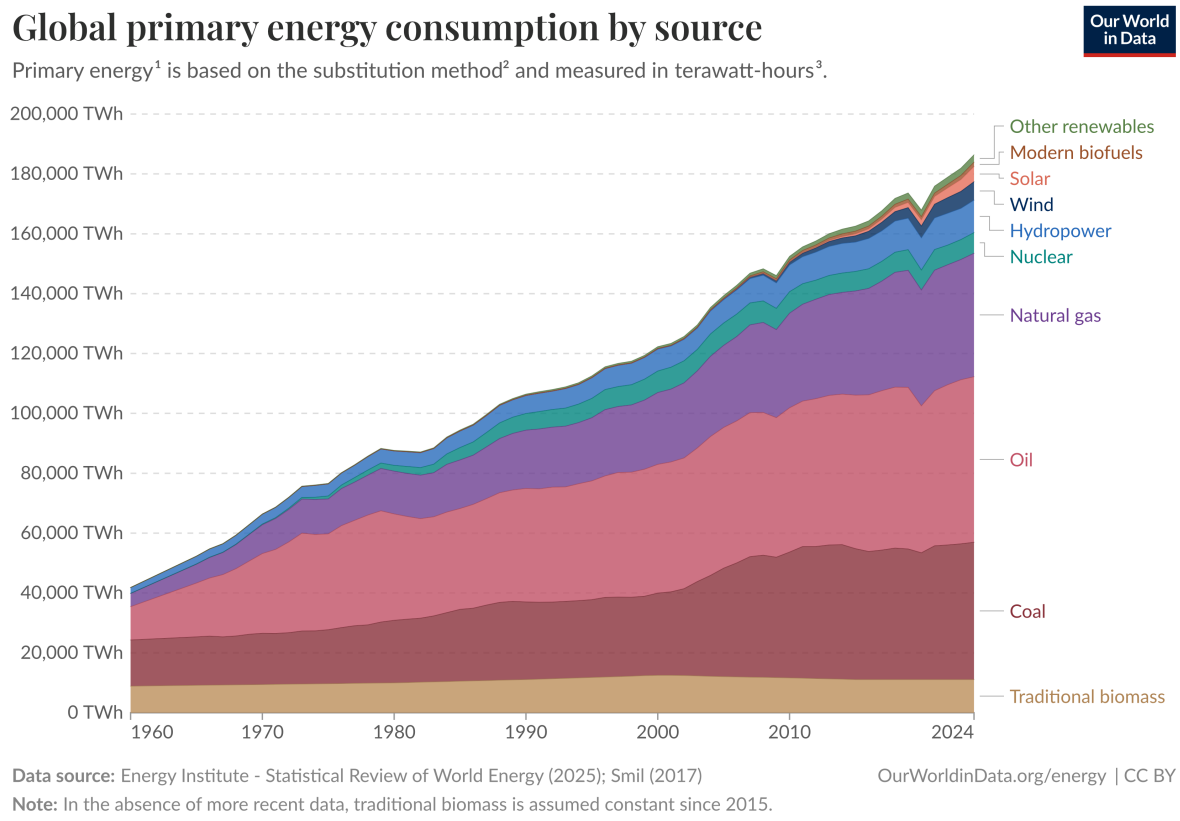

Figure 1.2. Global primary energy consumption by source from 1960 to 2024, measured in terawatt-hours (TWh) using the substitution method. Source: Energy Institute – *Statistical Review of World Energy* (2025); Smil (2017); processed and visualized by Our World in Data [6].

Bayesian Optimization (BO), which result in faster convergence toward steady states while maintaining high accuracy. The detection of this convergence, however, still remains an active challenge as PORTALS uses Gaussian Processes (GPs) as surrogate models which introduce a certain level of uncertainty to the problem.

The Ricci metric, offers a practical way to address the problem of convergence detection. However, its default parameters in PORTALS have been chosen heuristically which limits its interpretability and robustness across different plasma scenarios. By tuning the parameters that control the transition between agreement and disagreement regions, the Ricci metric can incorporate the inherent uncertainties of GPs and prove a consistent notion of convergence.

The main objective of this project is to improve convergence detection by integrating concepts from probabilistic modeling, BO, and plasma transport theory. The optimized Ricci metric parameters make the metric itself more consistent and objective. This reduces the reliance on manual inspection and improves reproducibility across different plasma configurations.

Ultimately, this research supports the goal of making fusion energy feasible and economically and environmentally viable.

1.3. Structure

The document is structured into six chapters, Chapter 1 gives a context about nuclear fusion and the motivation and purpose of the project. Chapter 2 reviews the state-of-the-art, giving a brief inquiry into fusion devices, surrogate modeling and the Ricci metric. Chapter 3 describes the methodology used for the project, including the dataset construction, labeling, and the optimization algorithm configuration. Chapter 4 presents and analyzes the results of the optimization and compares them to previous configurations. Chapter 5 provides a discussion of the findings, what are the implications of these findings, its limitations and a few ideas for future work. Lastly, Chapter 6 addresses the social and environmental impact of this work.

2. State of the art

2.1. Nuclear Fusion

Nuclear fusion is a nuclear process in which two or more nuclei combine to form a heavier nucleus. The energy produced in these process comes from the mass defect, the difference between the total mass of the initial nuclei and that of the fused nucleus (see Formula 2.1).

$$\delta = \text{Total nucleon mass} - \text{Nucleus mass} \tag{2.1}$$

When light nuclei like deuterium (^2H) and tritium (^3H) fuse to form a heavier nucleus, this resulting nucleus has slightly less mass than the sum of its parts. According to Einstein's equation $E = \Delta mc^2$ this mass difference is not lost, but transformed into energy. This energy is released in the form of kinetic energy of the reaction products, like alpha particles (^4He) and neutrons (see Figure 2.1.

This process is what happens at the heart of stars like the Sun which nucleus has temperatures of around 15 million $°C$. In order to recreate the conditions necessary for fusion to occur artificially and in a controlled fashion several confinement approaches have been studied, like inertial and magnetic confinement.

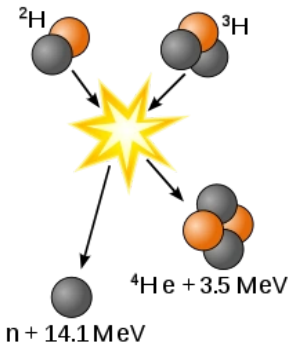

Figure 2.1. Fusion example of Deuterium and Tritium

2.2. Inertial Confinement

Inertial Confinement Fusion (ICF) is a process in which fuel particles are heated and compressed in order to achieve fusion. Typically a fuel pellet is rapidly heated and compress using powerful lasers or particle beams. This compression happens so quickly that the fuel is not able to expand and thus its own inertia keeps it confined long enough for fusion reactions to occur. During this process the outer shell of the pellet is blown off which creates an implosion and the fuel is compressed.

Thermonuclear devices like the hydrogen bomb are an illustrative example of ICF. In short, a first stage fission bomb is used to compress and ignite a secondary stage containing fusion fuel. Contrasting this, researchers attempt to harness the same physical principle in a controlled way.

ICF is a promising approach to fusion, as in December 2022 researchers at National Ignition Facility (NIF) achieved a fusion reaction that released approximately 3.15MJ of energy exceeding the 2.05MJ of laser energy delivered to the pellet, making it the first laboratory demonstration of a reaction in which positive net energy was produced [9].

2.3. Magnetic Confinement

In contrast with ICF, Magnetic Confinement Fusion (MCF) uses strong magnetic fields to confine plasma away from the reactor walls, maintaining the conditions required for fusion.

The most common MCF devices are tokamaks, stellarators and magnetic mirrors.

2.3.1. Tokamaks

The first proposal for using controlled fusion for industry use was formulated by Oleg Lavrentiev in a 1950 paper [10] and in 1954 the T-1, considered to be the first tokamak, was built in the Soviet Union.

Tokamaks, from the Russian 'тороидальная камера с магнитными катушками' (toroidal

chamber with magnetic coils) are donut-shaped devices used in MCF that combine toroidal and poloidal magnetic fields creating a twisted, helical magnetic field. The objective of the geometry is to confine the plasma away from the vessel walls, thus preventing energy loss in the form of heat and avoiding damaging the device. This configuration also reduces instabilities and improves plasma confinement, allowing the plasma to reach the temperature and density conditions necessary for fusion to occur. A representation of a cross-section of a tokamak can be seen in Figure 2.2.

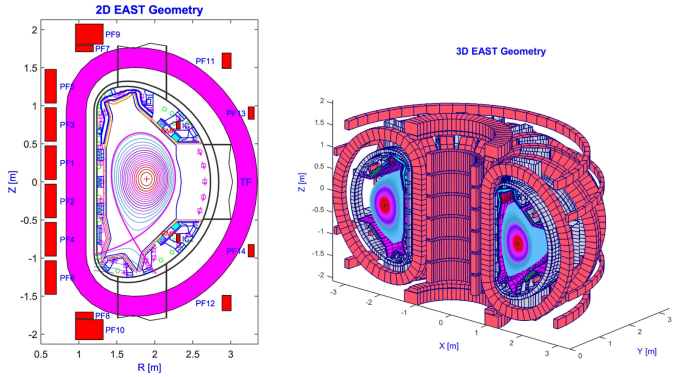

Figure 2.2. Illustration of tokamak geometries and 3D magnetic configuration. The toroidal shape of the confinement field allows for continuous plasma circulation under strong magnetic confinement.

These devices are currently the most widely studied in MCF due to their ability to maintain a plasma with those high temperatures and densities for long confinement times. The tokamak's helical magnetic field reduces particle drift and stabilizes instabilities that could disrupt the plasma otherwise.

Despite all the challenges, currently there are multiple tokamaks under development whose aim is to achieve what is known as the "scientific breakeven", producing more energy than that supplied to heat and sustain the plasma ($Q > 1$). Among these, the most prominent is International Thermonuclear Experimental Reactor (ITER) (see Figure 2.3), an international collaboration in France designed to produce 500MW of fusion power [11]. One of the key strategies ITER employs to achieve higher fusion power is increasing the plasma volume by enlarging the major radius of the tokamak. According to scaling laws, achieving ignition requires satisfying the Lawson criterion, which relates plasma density, temperature, and energy confinement time [12].

Meeting this condition allows the plasma to reach scientific breakeven, where the power generated by the fusion reaction is equal to the power used to heat and sustain the plasma. This threshold must be exceeded in order to achieve ignition, where plasma becomes self-sustaining

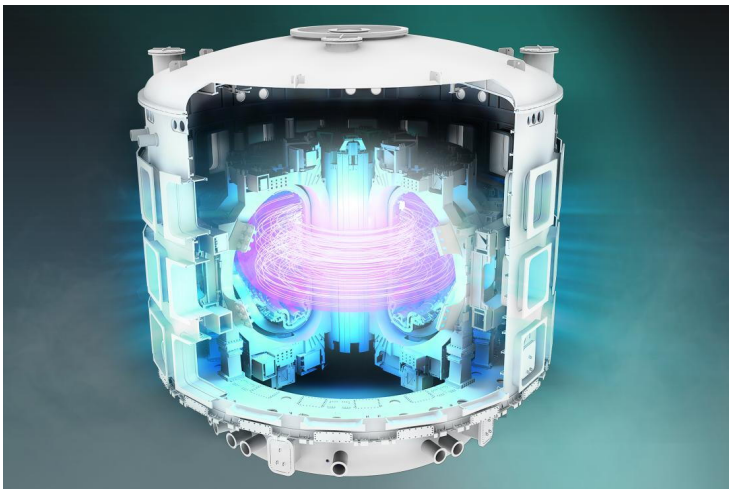

Figure 2.3. Computer-generated render of the ITER Tokamak

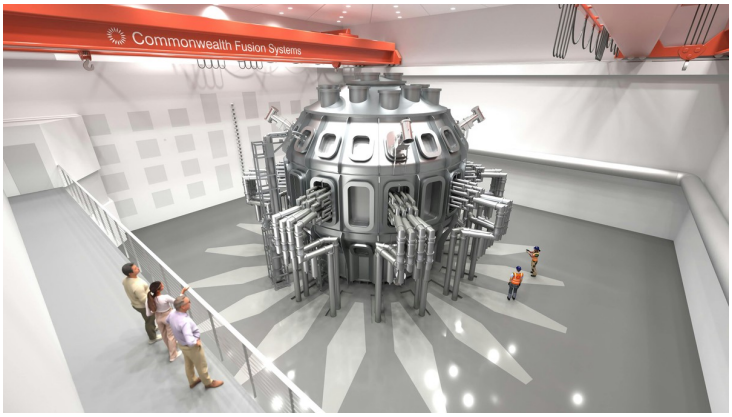

Figure 2.4. Computer-generated render of the SPARC tokamak

and there is no need for additional heating.

In contrast to ITER's approach, other devices like Smallest Possible ARC (SPARC) (see Figure 2.4) propose the opposite strategy. SPARC is a joint effort of the MIT's PSFC and the privately owned company Commonwealth Fusion Systems (CFS), it is a derivation of a previous device developed by the PSFC called Affordable, Robust, Compact (ARC). The developers of SPARC propose that rather than increasing the radius of the torus to confine more plasma, fusion conditions can be achieved in a much more compact device by creating a much stronger magnetic field using high-temperature superconductor magnets [13]. SPARC is currently being built at Devens, Massachusetts and it is projected to run its first plasma in 2026 and its stated goal is to demonstrate net energy production ($Q > 1$) by 2027.

Other experimental tokamaks such as DIII-D, developed by the U.S. Department of Energy, Office of Science [14], ASDEX-Upgrade in Germany and Alcator C-MOD developed at MIT

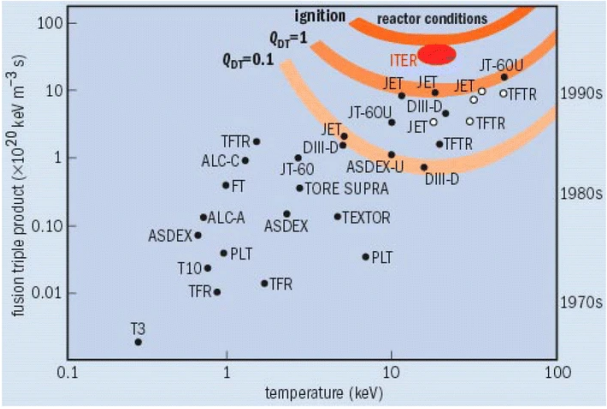

Figure 2.5. Progress of tokamak experiments in terms of the fusion triple product versus plasma temperature

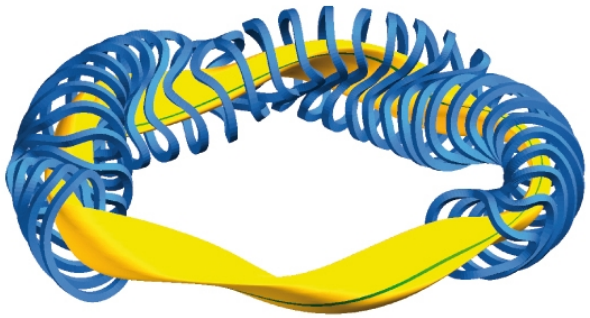

Figure 2.6. Computer-generated render of a stellarator

have also played a major role in advancing fusion research, paving the way for ITER and SPARC. This historical progression of tokamak experiments, summarized in Figure 2.5, illustrates the steady improvement in the fusion triple product toward breakeven conditions.

2.3.2. Stellarators

Regardless of these advantages, the complexity of building a stellarator is much greater than that of a tokamak, as it requires precisely shaped, three-dimensional magnetic coils and a highly optimized plasma vessel geometry. These components must be manufactured and assembled with extreme accuracy to ensure the magnetic field matches the theoretical design.

The most advanced stellarator to date is the Wendelstein 7-X, built in Greifswald, Germany in 2015 it has successfully demonstrated long-pulse plasma operation and confirmed the viability of optimized stellarator configurations. In Spain, the TJ-II stellarator, operated by the

CIEMAT in Madrid still remains an active experiment, highlighting that stellarators continue to be a promising research avenue for the future of fusion energy.

2.3.3. Magnetic Mirrors

Magnetic mirrors consist of a linear configuration where plasma is confined by strong magnetic fields that are stronger at both ends, this result in particles being reflected back into the central region. Magnetic mirrors suffer from significant particle loss, therefore limiting their performance. Despite this, they remain of scientific interest as they provide valuable insight into plasma confinement and stability.

2.4. The Role of Simulation

The design and operation of tokamaks or stellarators remains a challenge due to the nonlinear, multi-scale, and coupled nature of plasma behavior. Inside these reactors, the plasma is subject to turbulence, instabilities, collisional diffusion and external sources of heating, capturing these interactions with sufficient fidelity requires the use of complex and high-cost models that need to resolve microscopic and macroscopic processes simultaneously.

To predict and optimize plasma performance in these devices, physicists use transport solvers [15]. These are codes that evolve plasma profiles by iteratively solving the equations of particle, momentum, and energy balance. These solvers allow researchers to simulate how a plasma approaches steady-state under given operational conditions.

However, these transport simulations are notoriously computationally expensive, especially when using nonlinear gyrokinetic models, which can reach thousands of CPU hours for a single simulation. This challenge has motivated the adoption of alternatives based on surrogate modeling techniques to accelerate the transport simulation pipeline. These models can approximate the behavior of high-fidelity solvers while being significantly computationally cheaper. Furthermore, adding BO to these surrogates can accelerate the convergence toward steady-state solutions.

2.5. Transport Solvers

Traditional transport solvers such as *TRANSP* [16] or *ASTRA* [17] simulate the evolution of plasma quantities, such as electron and ion temperatures, density, and rotation. To achieve this, they iteratively solve the coupled balance equations for particle, momentum, and energy transport across flux surfaces.

These solvers use turbulence models like Trapped Gyro-Landau Fluid (TGLF) or Continuum Gyrokinetic (CGYRO) to compute the transport fluxes that appear in these equations. TGLF is a reduced quasilinear model that captures key gyrokinetic effects with low computational cost, which makes it perfect for rapid transport modeling [18]. And CGYRO is a nonlinear, first-principles gyrokinetic solver that provides high-fidelity calculations of turbulent transport in tokamak plasmas [19]. These models offer high physical fidelity, however, they are also the main computational bottleneck of the integrated modeling workflow. Each transport iteration executes tens to hundreds of gyrokinetic evaluations, and achieving a full steady-state convergence for electron temperature, ion temperature, and density can require up to thousands iterations. Furthermore, the coupling between turbulence, sources, and equilibrium evolution can cause the steady-state solution to depend sensitively on initial conditions or model parameters.

These limitations have motivated the development of more efficient alternatives such as surrogate-based methods, in which machine learning models are trained to emulate the behavior of expensive turbulence solvers. Once trained, these surrogates estimate transport fluxes, allowing the transport solver to evolve plasma profiles significantly faster.

Frameworks such as PORTALS use these surrogate models in combination with BO to guide the search towards a steady-state, therefore accelerating convergence and enabling uncertainty quantification within integrated plasma simulations.

2.6. The PORTALS Framework

A growing area of research in fusion simulation involves the use of surrogate models, data-driven approximations of computationally expensive simulations. Among these, GPs have become a standard due to their ability to capture uncertainty and provide smooth interpolations in low-data, high-cost regimes. The PORTALS framework uses GPs to model the behavior of transport fluxes as a function of plasma profile parameters, providing both predictions and uncertainty

estimates to guide the optimization process.

The main idea behind PORTALS is treating flux-matching as an optimization problem rather than relying on local Jacobian iterations. PORTALS trains GP surrogates of transport fluxes and uses their uncertainty-aware predictions to select the next simulations. This approach accelerates convergence to steady state and reduces the number of iterations for multi-channel predictions with gyrokinetic models.

The optimization is done by decomposing the residual into per-channel, per-radius components, effectively reducing the dimensionality of the problem. Then, uncertainty-aware GPs explore the best parameters to run the simulation with next, prioritizing informative evaluations while preserving the underlying physics.

Similar integrated modeling frameworks, such as OMFIT [20], have demonstrated how high-fidelity solvers and diagnostic data analysis can be orchestrated efficiently within large-scale fusion workflows.

2.6.1. PORTALS workflow

This subsection provides a conceptual overview of the PORTALS workflow, while specific implementation details and dataset construction steps are discussed later in Section 3.2.1.

The PORTALS workflow (Figure 2.7) is the following:

1. Initial Simulation Sampling: Run a small set of plasma transport simulations with random inputs[1].
2. Surrogate Model Construction: Train a GP on the collected data.
3. BO: Uses an acquisition function to select the best next most promising input.
4. Plasma simulation: Run a transport solver simulation using the parameters obtained in the previous step.
5. Convergence assessment: Check if the problem is converged, using the Ricci metric.
    - If the problem is converged proceed to step 6.
    - If the problem is NOT converged go to step 2.

[1]While these inputs are not strictly random, they are typically selected based on physical intuition. For simplicity, we assume them to be random in this description.

Figure 2.7. PORTALS workflow

6. Output solution: The optimal parameters obtained on the last iteration are returned.

2.6.2. Surrogate Modeling with Gaussian Processes

In PORTALS, GPs act as surrogate models that learn the mapping between plasma profile parameters and the resulting transport fluxes, providing both predictions and uncertainty estimates to guide the optimization. Their sample efficiency and calibrated uncertainty make them well suited for expensive simulations with few training points [21]. By leveraging these uncertainty-aware predictions, PORTALS can decide which simulations to run next, thereby reducing the number of costly evaluations on the high-fidelity model.

2.6.3. Bayesian Optimization

BO is a technique for optimizing black box functions without the need of gradients. BO uses a probabilistic model (usually a GP) of the objective function that is then used to search for the optimum. At each step the BO uses an acquisition function to decide where to sample next. There are a lot of different acquisition functions each one being used for exploration, exploitation or both, in this project the ones used are:

- Expected Improvement Monte Carlo (EIMC): Estimates the expected improvement over the best value seen so far, by sampling from the posterior distribution of the surrogate model. In the Monte Carlo version the expected improvement is computed by averaging multiple samples drawn from the GP.
- Logarithmic Expected Improvement Monte Carlo (LogEIMC): Similar to EIMC but applies a logarithmic transformation to the improvement.
- Posterior Mean: Always selects the point with the lowest predicted mean from the surrogate model.

- Simple Regret Monte Carlo: Estimates the simple regret, the difference between the true optimum and the predicted best point, by sampling from GP.

BO's use in physics is widespread due to their effectiveness for systems where the objective function is costly to evaluate and subject to noise [22]–[24]. BO, supported by GP surrogates, provides a robust and data-efficient strategy for optimizing computationally demanding plasma and fusion modeling workflows.

2.7. The Ricci metric

To determine whether the plasma has achieved a steady-state or not PORTALS makes use of the Ricci metric [5], a probabilistic measure that quantifies the discrepancy between two distributions while taking uncertainties into account (see Equation 2.2) . The Ricci metric plays a crucial role due to the inherent probabilistic nature of GPs, which provide not only predictions of transport fluxes but also associated uncertainty estimates. This allows the Ricci metric to evaluate convergence in a way that respects the confidence in each prediction, offering a more robust alternative to residual-based criteria.

$$\chi R = \frac{\sum_j R_j H_j S_j}{\sum_j H_j S_j} \tag{2.2}$$

Where:

- R_j: A soft agreement indicator, defined using a smoothed step function controlled by two tunable parameters d_0 and λ.
- H_j: A weighting factor inversely proportional to the radial mesh spacing, emphasizing the contribution of each radial point.
- S_j: A confidence weight that decreases when uncertainty is high relative to the flux magnitude.

The agreement term R_j is defined in Equation 2.3:

$$R_j = \frac{\tanh[(d_j - d_0)/\lambda] + 1}{2} \tag{2.3}$$

Where d_j (see Figure 2.4) represents the normalized discrepancy between the two measurements :

$$d_j = \sqrt{\frac{1}{N_j} \sum_{i=1}^{N_j} \frac{(x_{j,i} - y_{j,i})^2}{\Delta x_{j,i}^2 - \Delta y_{j,i}^2}} \tag{2.4}$$

The behavior of R_j, and therefore the overall Ricci metric, is controlled by two tunable parameters:

- d_0: Defines the threshold that distinguishes agreement from disagreement between the two measurements. The recommended ranges for do are $do \in [0.1, 2.0]$.
- λ: Controls the smoothness of the transition between agreement and disagreement. The recommended ranges for λ are $\lambda \in [0.1, 1.0]$.

2.8. Genetic Algorithms

GAs are algorithms inspired by natural selection and the theory of evolution. They were proposed in the 1970s by John Holland [25] as part of what he described as Evolutionary Algorithms (EAs) which simulated biological evolution to iteratively improve solutions to a defined problem.

2.8.1. Representation and Search Space

A GA works by operating on a population of N candidate solutions called individuals. Each of these individuals is encoded as a vector of m genes:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{im})$$

Each gene represents one tunable parameter of the problem and depending on the type of problem, genes can represent:

- Continuous parameters: They are typically represented as real numbers within predefined bounds. This is particularly useful for optimization problems that involver physical quantities or model hyperparameters.

- Integer or categorical variables: They are common in combinatorial problems such as scheduling or feature selection.
- Binary encoding: When GA were firstly proposed each gene was a bit that represented a decision variable.

The choice of encoding affects the choice of crossover and mutation operators, as different types of variables require different evolutionary strategies [26].

2.8.2. Selection Operators

Selection operators are used to determine which individuals of the population are chosen to contribute to the next generation. The goal of these operations is to balance exploration and exploitation. The most common selection operations are the following:

- Tournament selection: A random subset of individuals of size n is selected and the best of them is chosen.
- Roulette wheel selection: Assigns each individual a probability of being selected proportional to its fitness.
- Rank-based selection: Sorts individuals by fitness and assigns selection probabilities according to their rank rather than raw fitness values. This makes sure there is no premature convergence by maintaining diversity.

Tournament selection is the most widespread selection operation in practice due to its robustness and simplicity [27].

2.8.3. Crossover Operators

Crossover combines two parent individuals to produce offspring that inherit genes from both, the goal of this operators is to explore new regions of the search space.

The most common crossover operations are:

- Single-point and multi-point crossover: Parents exchange segments of their chromosomes at one or more crossover points, this is mostly used for discrete or binary encoding.

- Uniform crossover: Each gene in the offspring is randomly selected from either parent with equal probability.
- Simulated Binary Crossover (SBX): Proposed by Deb and Agrawal [28], SBX is a crossover operator that works on genes with real values instead of binary or discrete ones. Given two parent solutions, SBX samples a random value according a polynomial probability distribution to determine what is the intermediate value between them. The distribution is controlled by a parameter η_c that controls the shape of the probability distribution, a lower η_c will produce more variable offspring which will result in more exploration while a higher η_c will produce closer values to the parents.

2.8.4. Mutation Operators

Mutation introduces small random changes in one or more genes in order to maintain some genetic diversity and prevent premature convergence. It ensures that the algorithm continues to explore new areas of the search space even after starting convergence, this is useful to avoid local optima. The most common mutation types are:

- Bit-flip mutation: Used for binary encoding, where a bit is flipped with a small probability p_m.
- Gaussian mutation: Adds a normally distributed random value to each gene.
- Polynomial Mutation (PM): Proposed by Deb and Goyal [29], it consists of a self-adaptive operator for continuous domains that perturbs genes according to a polynomial distribution controlled by a parameter η_{mut}.

Self-adaptive mutation

There is an alternative strategy to fixed mutation parameters called self-adaptive mutation, in which the mutation rate evolves with the population. Each individual also contains mutation parameters that evolve overtime being subject to their own selection and mutation, by doing this the algorithm can control the intensity of the exploration, which usually increases mutation early on and the reduces it as it gets near to being converged [30].

2.8.5. Elitism and Termination Criteria

Elitism is the mechanism the GA uses to preserve individuals from one generation to the next, thus guaranteeing that the overall population fitness does not deteriorate over time. The vast majority of modern GA implementations, including the one used in this project (`pymoo`), maintain at least one elite individual per generation.

The algorithm continues evolving until a termination criterion is met. The most typical termination criteria are:

- A fixed number of generations (n_{gen}).
- Convergence of the population (e.g., low variance in fitness).
- Achievement of a target fitness value.

2.8.6. Fitness Design

The fitness function is essential for the GA as it is what gives us a notion of the performance of a solution. It works by assigning each individual in the population a value proportional to its quality, thus defining how the algorithm will search for optimal solutions.

In order to consider a fitness function effective it needs to satisfy a few key properties. First, the function has to be aligned with the objective of the optimization, making sure that any improvement in the fitness is correlated in some way to progress toward the desired solution. Second, it should be sensitive enough to be able to differentiate between individuals of different quality but not too sensitive so it is not affected by noise. Lastly, GAs usually require a large amount of fitness evaluations so the fitness function should be computationally efficient to evaluate.

There are multiple ways to define a fitness function depending on the problem. If the problem has only one objective, the fitness is usually the objective function value itself, possibly normalized or scaled to improve stability. However, if the problem has multiple objectives, the fitness requires a balance between competing goals, which is usually achieved through weighted aggregation or Pareto dominance ranking [26].

Smooth fitness functions tend to lead to faster convergence while rugged ones tend to get stuck at local optima. Therefore, in order to prevent the GA from getting stuck in local optima or

converging prematurely, the choice of the parameters for the selection, mutation and crossover has to be done while taking the type of fitness function into account [27].

2.8.7. Constraint Handling

It is not uncommon to have constraints in optimization problems, some candidates may violate constraints that make them unable to work as a solution or just make them a worse solution overall. There are several strategies the GAs tackle this problem, the most common are the following [26], [31]:

- Penalty methods: Adds a penalty to the fitness function proportionally to how much the constraint was violated or how critical the violation is.
- Repair operators: Modify individuals that violate a constraint to make them valid again, for example by changing their values.
- Selection based on feasibility: The GA prefers feasible individuals even if their fitness is worse, making sure that valid solutions dominate the population over time.

3. Methodology

3.1. Problem Definition

The goal of this project is to tune the Ricci metric parameters so it can better represent the convergence status of the plasma simulations. The problem can be expressed as a minimization like:

$$\min_{d_0,\lambda} \mathcal{L}(d_0, \lambda) \tag{3.1}$$

where $\mathcal{L}$ is the penalized RMSE between the convergence iteration detected by the Ricci metric and the true convergence iteration labeled in the dataset. As described in Section 2.7 the parameters (d_0, λ) represent the threshold between agreement and disagreement, and the smoothness of this transition respectively. By finding their optimal values one can make sure that the Ricci metric is sensitive enough to detect convergence early but robust enough to avoid false positives.

The Ricci metric itself depends on values that come from surrogate models and simulation data which makes it inherently noisy and stochastic. Additionally, the relationship between the Ricci parameters (d_0, λ) and the resulting metric behavior is non-linear and non-convex, this makes deterministic optimization methods or methods based on gradients highly ineffective.

Other optimizations strategies like BO or Particle Swarm Optimization (PSO) were considered but finally GAs were chosen due to their robustness to noise and its ability to balance both exploration and exploitation [26]. The GA starts by creating a population of candidate solutions which allows the sampling of multiple regions of the search space at once and crossover and mutation operations promote diversity, ensuring that the algorithm can continue discovering promising areas even after many generations and avoid local optima.

Another reason why the GA was chosen is that when compared to other global optimization techniques, GAs offer several advantages for this problem. For example, PSO tends to converge rapidly but in doing so it may lose diversity in search spaces as small as the one defined by our two Ricci parameters. Another example is BO, which despite being highly effective for smooth

and expensive black-box functions, introduces a significant computational overhead as it uses GPs, which is unnecessary for a problem that has fairly inexpensive evaluations. In contrast, GAs is perfect for the use case of the project, it provides a solution that can efficiently explore continuous search spaces and still be robust to stochastic variations in the fitness function.

For all these reasons, that an evolutionary approach based on a GA was adopted to optimize the Ricci metric parameters.

3.2. Dataset Construction

3.2.1. Simulation data

The dataset used in this project was created from a total of 19 independent PORTALS simulations, each one the simulations consist of an iterative optimization loop that uses a transport solver guided by surrogate models and BO. Of these 19 simulations only 13 of them were convergent, their convergence time can be seen in Table 3.1.

Table 3.1. Convergence summary for all plasma simulations. The table shows whether each case reached convergence and the iteration T at which it occurred.

Plasma	**Converged**	**Iteration** T
AUG_eimc	Yes	13
AUG_logeimc	Yes	12
AUG_posteriormean	Yes	12
AUG_simpleregretmc	Yes	13
CMOD_eimc	No	–
CMOD_logeimc	No	–
CMOD_posteriormean	No	–
CMOD_simpleregretmc	No	–
ITER_logeimc	Yes	11
ITER_posteriormean	Yes	11
ITER_simpleregretmc	Yes	12
SPARC-LMODE_eimc	Yes	18
SPARC-LMODE_logeimc	Yes	25
SPARC-LMODE_posteriormean	Yes	15
SPARC-LMODE_simpleregretmc	Yes	15
SPARC-PRD_eimc	Yes	29
SPARC-PRD_logeimc	No	–
SPARC-PRD_posteriormean	Yes	30
SPARC-PRD_simpleregretmc	No	–

During the first iteration of the simulation, PORTALS saves a full snapshot of the optimization in a *.pkl* file that is then updated for each iteration. This object contains the surrogate models, acquisition values, and simulation outputs for each iteration. For this project only the powerstates of the object were extracted, as they contain the target and transport fluxes, which are the variables required to compute the Ricci metric.

However, reading the *.pkl* file containing the powerstates is really slow, for that reason, a preprocessing script (`extract_data.py`) was developed to iterate through all the simulations, extract relevant quantities, and save them into lightweight *.json* files with the following format:

```
{
  "AUG_eimc": [
    {
```

```
      "y1": [...],
      "y2": [...],
      "y1_std": [...],
      "y2_std": [...]
    },
    ...
  ]
}
```

Here *AUG* refers to the ASDEX Upgrade tokamak, while *eimc* is the acquisition function used in that simulation. The variable *y1* corresponds to the simulated transport flux and *y1_std* its uncertainty, while *y2* and *y2_std* correspond to the target flux and its uncertainty, respectively.

The dataset includes simulations of different plasma configurations and devices, each with distinct transport characteristics and convergence behaviors. This diversity ensures that the optimization of the Ricci parameters does not overfit to a particular machine or operational regime.

3.2.2. Data labeling

Each simulation was manually inspected to determine when it reached convergence or if it did reach it at all. For those that did, the iteration at which convergence occurred was labeled using the field `converged_at`. Simulations that never converged were labeled with `null`.

Despite this technique being subjective it establishes a baseline against which to assess the Ricci metric detection.

To evaluate the generalization capability of the optimized Ricci parameters, a validation strategy inspired by machine learning was employed, the available simulations were divided into K folds, and the GA was trained on $K-1$ while being evaluated on the fold left out of the training subset. Cross-validation (see Section 3.5) is a common strategy in traditional optimization problems, and its use is justified in this context because the goal is not to optimize each simulation independently, but rather to find global parameters that generalize well across different plasma configurations. This approach measures how well the Ricci metric performs on unseen data and helps keep the algorithm from memorizing the dataset.

To generate these folds, the `stratify_data.py` script was developed. This script loads the

extracted simulation data and organizes it into stratified folds, ensuring that both convergent and non-convergent cases are evenly distributed across them. Each plasma entry is stored together with its corresponding `converged_at` label and iteration data, enabling consistent and reproducible evaluation of the Ricci metric calibration across multiple training and testing splits. The resulting *.json* file has the following structure:

```
{
  "0":
    {
      "AUG_eimc" : {
        "converged_at" : 13,
        "iter_list" : [
          {
            "y1": [...],
            "y2": [...],
            "y1_std": [...],
            "y2_std": [...]
          },
          ...
        ]
      },
      ...
    },
  ...
}
```

The top level keys (`"0"`, `"1"`, `"2"`, ...) are the individual fold, each fold contains a key (e.g., `"AUG_eimc"`) that refers to an individual plasma, in this example, the ASDEX Upgrade tokamak using the *eimc* acquisition function.

For each plasma entry, the field `"converged_at"` contains the iteration number at which convergence was observed (or `null` if the case had no convergence) and `"iter_list"` contains the full time series of iteration data. Each element in `"iter_list"` is a dictionary with the transport flux (`y1`), its uncertainty (`y1_std`), the target flux (`y2`), and its uncertainty (`y2_std`).

3.3. Fitness Function

The fitness function measures how well a pair of (d_0, λ) parameters allows the Ricci metric to detect convergence. If the detection is done too early it can misrepresent a plasma as converged when it still has not converged yet, but if it is done too late it can result in more expensive transport simulations which is computationally inefficient. This motivated the inclusion of a constraint to account for incorrectly detected convergence.

It is critical to design an appropriate fitness function as a smooth or uninformative fitness can slow convergence and a steep one can lead to instability and premature convergence as explained in Subsection 2.8.6.

For each simulation, the Ricci metric $\chi_R(t)$ is computed for all iterations and the first iteration where the $\chi_R(t)$ is below the threshold of $\varepsilon = 0.05$ is considered as the convergence point. The value of $\varepsilon = 0.05$ was selected after experiments comparing values in $\{0.01, 0.05, 0.1, 0.2\}$. These experiments confirmed that if the threshold was too low some cases would get stagnated without converging, while higher thresholds produced excessive false positives. The chosen value had the most consistent results being both sensitive and robust enough for the problem. Additionally, in some experiments ε was added as a tunable parameter of the GA but the results of the optimization showed no significant improvement in the overall fitness or generalization performance.

The fitness function distinguishes these two cases:

- Convergent simulations: The fitness penalizes the squared distance between the detected and true convergence iterations. Early detections ($t_{\text{detected}} < t_{\text{real}}$) are penalized twice as much because identifying convergence prematurely implies a loss of physical accuracy.
- Non-convergent simulations: If χ_R crosses the threshold erroneously, a large penalty is applied, as this corresponds to a false positive detection.

The final fitness value is defined as the square root of the average penalized error, as expressed in Equation 3.2.

$$\text{Penalized RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\begin{cases}(P^2, & \text{if } t^{(i)}_{\text{detected}} \text{ not found} \\ 2(t^{(i)}_{\text{real}} - t^{(i)}_{\text{detected}})^2, & t^{(i)}_{\text{detected}} < t^{(i)}_{\text{real}} \\ (t^{(i)}_{\text{detected}} - t^{(i)}_{\text{real}})^2, & \text{otherwise}\end{cases}} \tag{3.2}$$

Where P is penalty constant defined for the problem and N denotes the total number of plasma simulations considered in the dataset. Simulations without a convergence label are excluded from this average but still influence training through penalty assignment when necessary.

The idea is that the penalized RMSE aggregates results from all the simulations, thus ensuring that the optimization does not overfit to a single plasma case and instead it finds (d_0, λ) values that generalize well across different configurations.

Figure 3.1 illustrates the workflow of the fitness evaluation process. Each candidate pair (d_0, λ) is evaluated by computing the Ricci score for all iterations in each simulation. From this time series, the predicted convergence iteration t_{detected} is extracted, compared to the ground-truth label t_{real}, and transformed into a penalized error. The penalized RMSE of all simulations is the combined and is considered the final fitness value for that individual.

3.4. Genetic Algorithm Setup

The objective of the GA is to minimize the penalized RMSE defined in Equation 3.2 by tuning the parameters (d_0, λ) of the Ricci metric. The solution space of the problem is continuous, with $d_0 \in [0.1, 2.0]$ and $\lambda \in [0.1, 1.0]$, which are the recommended ranges in the original Ricci metric formulation [5].

Each individual in the population represents a pair (d_0, λ) encoded as a vector of real values. At each generation, the population evolves through the genetic operators described in Section 2.8. Specifically, tournament selection was used to determine the individuals contributing to the next generation. For crossover, the SBX operator was used, with a probability of $p_c = 0.9$ and a distribution parameter of $\eta_{cx} = 15$. For mutation, PM was used with $\eta_{mut} = 20$ and the mutation probability p_m being adaptively controlled by `pymoo` based on the dimensionality of the problem.

Elitism is applied to the problem to ensure that the best performing individuals end up in the following generation making sure the fitness does not decrease between iterations.

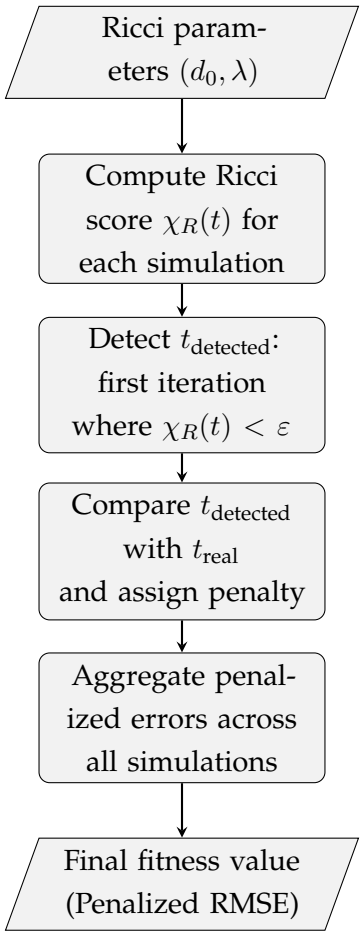

Figure 3.1. Workflow of the fitness evaluation process. Each candidate (d_0, λ) is used to compute the Ricci score for all simulations. The detected convergence iteration is compared to the labeled one, and the resulting penalized RMSE determines the individual's fitness.

3.5. Cross-Validation and Experimental Protocol

To evaluate how well the GA could generalize the parameters for the Ricci metric a K-fold cross-validation strategy was used. As described in Section 3.2.1, the available simulations were partitioned into $K = 5$ stratified folds, this means that every fold contains a balanced distribution of convergent and non-convergent cases. Since only 19 plasma simulations were available, using a larger number of folds would have produced training sets that were too small and unrepresentative, while a smaller K would have reduced the robustness of the validation.

For each run, the GA was trained on the data of $K - 1$ folds and validated on the remaining one.

This process is repeated K times ensuring that every fold has been used once as validation data, and the performance metrics are averaged across all folds.

For every fold, the GA was initialized with a population of 30 individuals and evolved for 20 generations and each fold was executed 32 times with different random seeds to account for the stochastic variations of the algorithm. The choice of 32 runs per fold was made to ensure a statistically reliable estimation of the algorithm's performance while keeping the computational cost assumable.

The best individuals of each run, their fitness values and their corresponding (d_0, λ) parameters were saved and stored in a `.json` file with the following structure:


```
{
  "timestamp": "20250702_125338",
  "K": 5,
  "N": 32,
  "stratify_seed": 2825,
  "algorithm_seed": 2708,
  "folds": [
    {
      "fold": 0,
      "runs": [
        {
          "d0": 1.9584297046460606,
          "lambda": 0.23204887989510162,
          "evaluation": 1.9148542155126762,
          "seed": 2708
        },
        ...
      ]
    },
    ...
  ]
}
```


The top level fields define the global configuration of the experiment, number of folds (`K`), the total number of independent runs (`N`) and the seeds used for each part of the experiment (`stratify_seeda` and `algorithm_seed`).

Then, `"folds"`, is a list containing one entry per cross-validation fold. Each fold entry includes a list of `"runs"`, where each run stores the optimized Ricci parameters (`d0` and `lambda`) obtained by the GA and their corresponding fitness value (`evaluation`).

All scripts used in this work are publicly available at:
https://github.com/amsanabria/ricci-metric-optimization

4. Results

This chapter presents the results of the Ricci metric's optimization. First the experimental setup and dataset partitioning used for the cross validation is described, then these results are analyzed in order to assess their validity. In the following sections an evaluation of the generalization performance across folds will be done and the optimized configuration will be compared against the baseline and manually tuned versions used in PORTALS.

4.1. Overview of Experiments

In order to evaluate the proposed Ricci metric parameters, a series of experiments were conducted using the dataset described in Section 3.2.1.

A K-fold cross-validation with $K = 5$ was done to try to minimize the impact of the training data in the final result by ensuring that every subset of simulations was used once for testing and four times for training. For each experiment corresponding to a given fold, the data were partitioned so that one subset (e.g., Fold 0) was held out as the test set, while the remaining four folds (e.g., Folds 1, 2, 3 and 4) were used for training. This means that the genetic algorithm was optimized using only the training folds, where the fitness function evaluated performance on those data, and the resulting parameters (d_0, λ) were later validated on the held-out test fold to assess generalization. For each fold, the genetic algorithm was executed 32 times with different random seeds to account for stochasticity. This resulted in a total of 160 independent runs, whose results were used to analyze parameter stability and algorithmic robustness.

Experiments were performed using the configuration and fitness function for the genetic algorithm outlined in Section 3.4. This is a consistent setup to ensure that variability between the folds can be assigned mostly to difference in the data instead of modification in the optimization process.

4.2. Implementation Details

The algorithm was implemented using the `pymoo` library [32], a Python framework that offers state of the art optimization algorithms.

The chosen hyperparameters follow standard recommendations for continuous-domain genetic algorithms [26]. A high crossover probability ($p_c = 0.9$) promotes exploration of the search space, while the distribution parameters $\eta_{cx} = 15$ and $\eta_{mut} = 20$ ensure smooth variation around parent solutions without excessive disruption. The population size (30) and number of generations (20) were empirically selected to balance convergence stability and computational efficiency.

The configuration parameters used in all experiments can be seen in Table 4.1.

Table 4.1. Configuration of the genetic algorithm and fitness parameters used in all experiments.

Parameter	**Value / Description**
Population size (pop_size)	30
Number of generations (n_gen)	20
Crossover operator	SBX
Crossover probability (p_c)	0.9
Crossover parameter (η_{cx})	15
Mutation operator	PM
Mutation probability (p_m)	Adaptive
Mutation parameter (η_{mut})	20
Sampling method	Latin Hypercube Sampling (LHS)
Number of runs per fold	32
Number of folds (cross-validation)	5
Fitness threshold (ε)	0.05
Penalty constant (P)	20

The initial population is generated using LHS, this is a method that generates parameter combinations that are evenly distributed in the search space, leading to a more representative initial population. Then, the population evolves over a number of fixed iterations and each individual is evaluated by the function described in Section 3.3. The optimization runs for $n\_gen = 20$ generations with a population size of $pop\_size = 30$ individuals. At the end of each generation,

there are a total of 30 parents + 30 offspring = 60 individuals. From this combined population, only the best 30 are selected to survive into the next generation, this maintains the defined population size.

The penalty constant is set to 20 in order to provide a fair balance between penalizing incorrect or missing convergence detections and avoiding excessive bias in the fitness. It was empirically chosen as it is a high enough number to make the algorithm avoid false positives but not so high that it dominates the overall error or prevents the exploration of slightly imperfect yet informative solutions.

4.3. Optimization Results

The optimization process was executed 32 times for each of the 5 folds with randomized seeds, resulting in 160 runs. Each of these runs returned a pair of values for d_0 and λ as well as a fitness value representing the penalized RMSE of the training set.

Figure 4.1 shows the relationship between the Ricci metric parameters (d_0, λ) and the fitness values across all optimization runs. Most of the points are clustered inside $d_0 \in [1.94, 2.0]$ and $\lambda \in [0.1, 0.4]$ and their error is fairly low, as denoted by the lighter tones, however, the points outside this region, all of which happen to be from Fold 0, exhibit a higher error, this phenomenon is more thoroughly explored in Section 4.4.

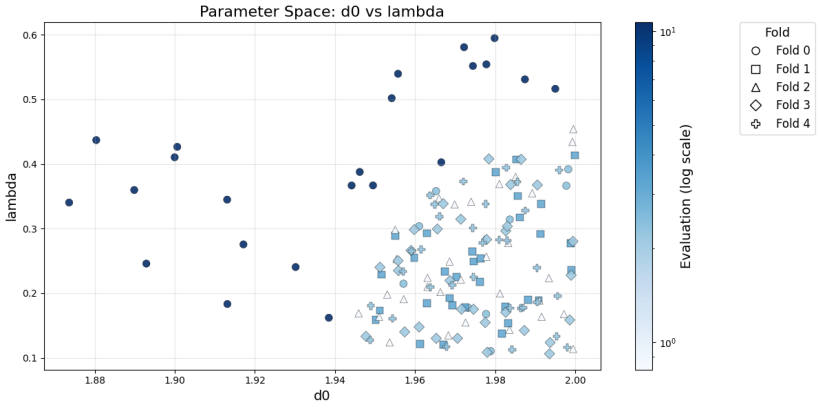

Figure 4.1. Correlation between Ricci metric parameters (d_0, λ) and fitness values across all runs. Lighter regions correspond to lower penalized RMSE (better), highlighting the stable optimum near $(d_0 \approx 1.97, \lambda \approx 0.17)$.

The mean value of d_0 for all runs was approximately 1.97 ± 0.02, while λ averaged 0.17 ± 0.08. Most of the solutions are centered around this region, indicating that the optimization tends to give d_0 high values while doing the opposite for λ.

The histograms in Figure 4.2 show most of the values being concentrated in values near the upper limit of the search range for d_0 ($d_0 \approx 1.97 \pm 0.02$), while λ values exhibit a broader distribution centered around 0.25 with some values being higher up to 0.6.

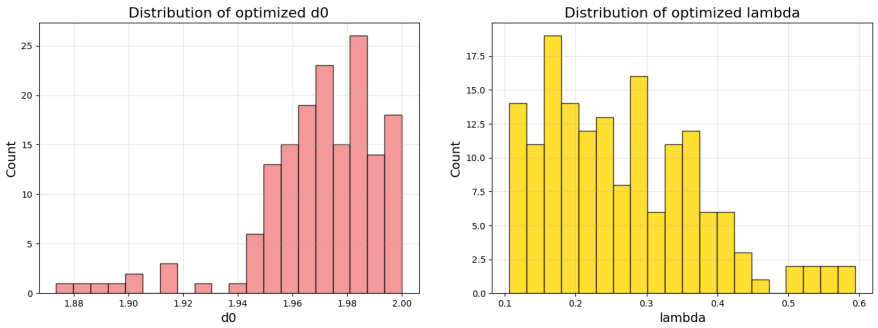

Figure 4.2. Aggregated distributions of the optimized d_0 and λ parameters across all folds.

Table 4.2 shows the mean and standard deviation of the final fitness values for each fold.

Table 4.2. Average and standard deviation of the final fitness values obtained in each fold.

Fold	Mean Fitness	Std. Dev.
0	8.332	3.783
1	3.109	0.000
2	0.816	0.000
3	2.121	0.000
4	2.236	0.000

Folds 1, 2, 3, and 4 have the lowest average error, this may indicate that their corresponding training sets produced consistent and well-calibrated Ricci parameters. Contrasting this, Fold 0 shows a higher mean error, which may indicate that it contains more heterogeneous or difficult plasma cases that challenged the optimization or the validation process.

In addition to this, Fold 2 achieved a significantly lower mean fitness value (0.816) in comparison to the other folds, despite having similar parameter distributions to Folds 1,3 and 4 (as seen in Figure 4.1). This could be explained by a more favorable composition of training and validation simulations, allowing the GA to converge more efficiently and reach near-optimal (d_0, λ)

combinations with minimal penalization. The imbalances seen in Folds 0 and 2 could indicate a potential imbalance in the cross-validation partitioning, where certain folds contained easier or less variable cases while others contained more difficult or variable plasmas.

In order to understand these discrepancies of the experiments a dissection of each fold was performed to confirm which plasmas were in which fold, the composition of the folds can be seen in 4.3. The analysis reveals that some folds contained a larger proportion of certain configurations, for example, Fold 0 includes multiple AUG and SPARC-PRD cases, while Fold 2 contains several SPARC-LMODE and ITER simulations, which could explain its inconsistency with the rest of the folds.

Table 4.3. Composition of the 5 folds used in the cross-validation procedure. Each fold contains the plasma simulations assigned during stratification.

Fold	Plasma simulations included
0	AUG_posteriormean
	AUG_logeimc
	AUG_simpleregretmc
	SPARC-PRD_simpleregretmc
	CMOD_eimc
1	ITER_posteriormean
	SPARC-LMODE_logeimc
	SPARC-PRD_posteriormean
	CMOD_posteriormean
2	SPARC-LMODE_eimc
	SPARC-LMODE_simpleregretmc
	ITER_simpleregretmc
	CMOD_logeimc
3	SPARC-PRD_eimc
	ITER_logeimc
	CMOD_simpleregretmc
4	AUG_eimc
	SPARC-LMODE_posteriormean
	SPARC-PRD_logeimc

4.4. Cross-Validation Performance

As explained in Subsection 3.2.1 a 5-fold cross-validation procedure was done in order to evaluate the generalization ability of the optimized Ricci metric parameters. Each fold was treated as an independent experiment in which the genetic algorithm was trained using 80% of the dataset, while the remaining 20% were used for validation. This approach ensures that every plasma case is used for both training and testing at least once, providing a robust estimate of the Ricci metric's generalization abilities.

In Figure 4.3 the distribution of the final fitness values obtained across all folds is shown. The optimized Ricci metric achieved, on average, a test fitness of approximately 2.9 ± 3.1. However, the distribution has a clear outlier, Fold 0 shows higher variability and error values (going up to $\sim$ 10), while the other folds maintain consistent fitness values between 1 and 3. This may indicate that the optimization performs relatively well for most data partitions but is sensitive to certain subsets, this being mostly likely to differences in plasma configurations or surrogate model behavior included in Fold 0.

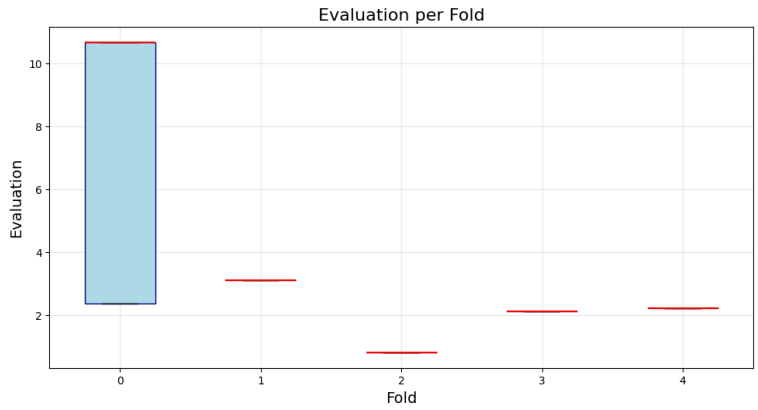

Figure 4.3. Distribution of penalized fitness values obtained for each fold during cross-validation.

These results show a consistent error range for most of the folds, with Fold 0 having worse (higher) fitness values compared to the rest of folds. While Folds 1 to 4 have relatively stable and low Penalized RMSE values (between 0.8 and 3.1), Fold 0 shows values going from 2.5 to 10.5, clearly standing out as an outlier.

A deeper analysis of Fold 0 revealed that its poor performance is mainly due to the presence of two particularly challenging plasma cases: `SPARC-PRD_simpleregretmc` and `CMOD_eimc`.

Both simulations are labeled as non-convergent. This also happens in other folds, however, it is possible that the combination of these two specially challenging cases may have resulted in various GA runs in which they were no valid points, i.e. during the whole fitness calculation process not a single case was detected as converged. This is impossible as at least one converged case is guaranteed to be converged, so the fitness applies a heavy punishment considering the solution provided invalid.

In order to better understand the values of the rest of the folds an auxiliary figure was created in which the boxplot is done without the information of Fold 0, these results can be seen in Figure 4.4. The figure shows that there are no boxes this is due to all of the executions of the GA converging to the same value. This absence of variability is not due to a lack of diversity in the GA itself, but rather to the discrete nature of the fitness function (see 3.2). Since the Ricci metric detects convergence at specific iteration indices (t_detected and t_real), the resulting penalized RMSE values are computed from integer differences between iterations. This results in the fitness being quantized, taking only a small number of possible values.

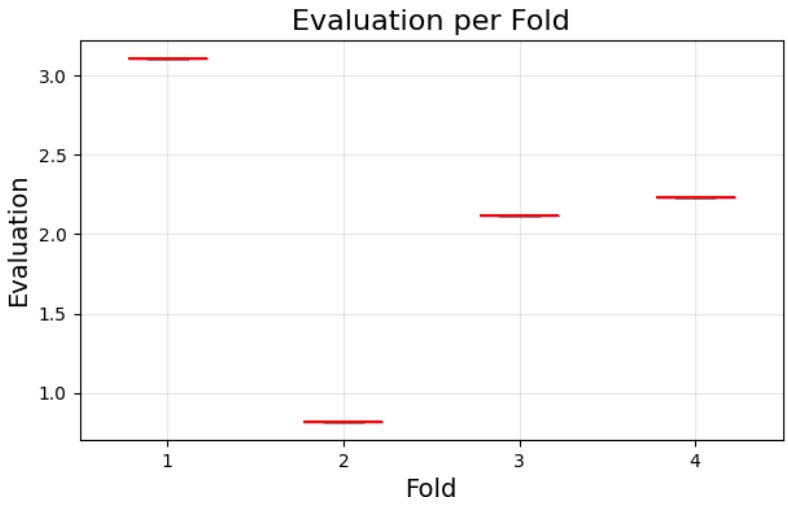

Figure 4.4. Penalized fitness distribution for Folds 1–4, excluding Fold 0. The reduced range highlights the stability of the GA performance across the remaining folds, with values consistently between 0.8 and 3.1.

To verify whether the differences observed among folds were significant or not, a Friedman test was conducted using the fitness values of each fold. The Friedman test is a non-parametric statistical test used to detect differences between experiments [33]. The test returned a χ^2 of 122.83 with a p-value of 1.33×10^{-25}, which was below the $\alpha = 0.05$ threshold, this meaning that the differences across folds are statistically significant.

Once the Friedman test was done, a post-hoc Nemenyi test [34] was conducted to identify which

pairs of folds exhibited statistically significant differences. This test identified multiple significant differences, especially between Fold 0 and the remaining folds, as well as between Fold 1 and the Folds 2 and 3. The critical difference was 1.52, and the lowest average rank was achieved by Fold 2, followed by Fold 3 and Fold 4, while Fold 0 was the worst (see Figure 4.5). These confirmed what we could see in Figure 4.3, Fold 0 is an outlier, while the other folds are homogeneous.

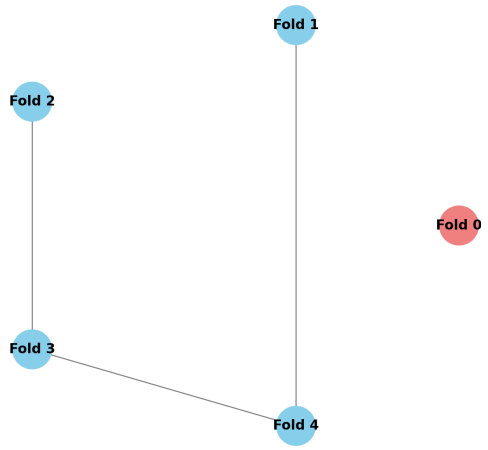

Figure 4.5. Graphical representation of the Nemenyi post-hoc test results. Each node corresponds to a cross-validation fold, and edges connect pairs of folds whose performance differences are *not statistically significant* (i.e., $|\Delta\text{rank}| \leq CD = 1.52$). Fold 0 appears isolated, confirming its significantly worse performance, while Folds 2, 3, and 4 form a connected group, indicating statistically similar behavior.

4.5. Comparison with Baseline Ricci Metric

The performance of the calibrated Ricci metric was compared against two reference configurations: the *baseline* parameters used in the PORTALS software as default [4] ($d_0 = 2.0$, $\lambda = 1.0$), and a manually tuned configuration ($d_0 = 2.0$, $\lambda = 0.5$) obtained through empirical inspection of several simulations. Both configurations were evaluated under the same conditions as the optimized one, using the same dataset and cross-validation setup.

Table 4.4 summarizes the average fitness achieved by each configuration for the five folds. The optimized configuration obtained an average penalized RMSE of 2.13 ± 0.74, while the manual and baseline settings reached 3.79 ± 3.51 and 8.32 ± 5.16, respectively. The optimized version consistently achieved the lowest error of all of the folds, indicating that the genetic algorithm

was successful in providing a set of parameters that better capture the convergence behavior.

Table 4.4. Comparison of the average Penalized RMSE obtained for the baseline, manual, and optimized Ricci configurations across all folds.

Configuration	d_0	λ	**Mean RMSE $\pm$ Std. Dev.**
Baseline	2.0	1.0	8.3244 ± 5.1611
Manual	2.0	0.5	3.7889 ± 3.5134
Optimized (GA)	1.971	0.168	2.1327 ± 0.7427

Figure 4.6 gives a graphical comparison of the average test errors for each configuration. The optimized Ricci metric outperforms the baseline and manual versions, showing a 74.38% reduction in the mean error compared to the baseline, and a 43.71% reduction compared to the manual configuration.

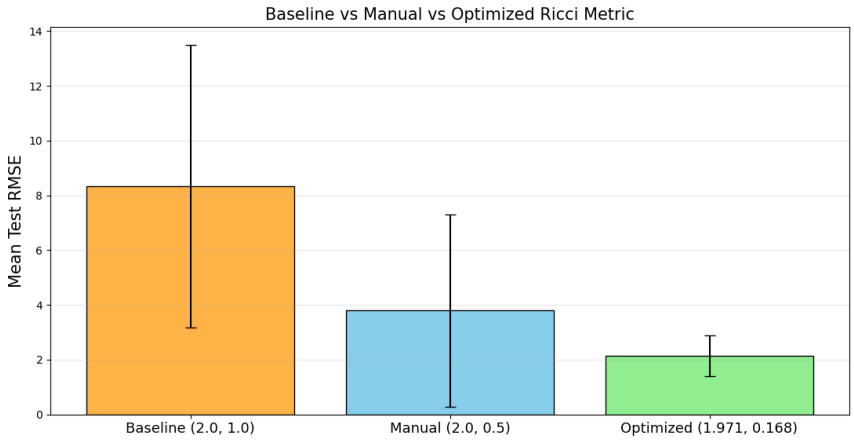

Figure 4.6. Mean test Penalized RMSE across folds for the baseline ($d_0 = 2.0$, $\lambda = 1.0$), manual ($d_0 = 2.0$, $\lambda = 0.5$), and optimized Ricci configurations. The colored bars represent the mean performance across cross-validation folds, while the vertical black lines indicate the standard deviation of the test RMSE, reflecting variability across folds. Lower bars and shorter error lines denote better and more consistent convergence detection.

To visualize the impact of the parameter tuning, Figure 4.7 shows the evolution of the Ricci score $\chi_R(t)$ for a sample plasma simulation (*ITER_logeimc*). The optimized configuration (in green) reaches the convergence threshold $\varepsilon = 0.05$ two iterations earlier than the baseline (in blue). The manually tuned configuration (in orange) also improves upon the baseline, crossing the threshold at the same iteration as the optimized one. These results highlight that the optimized parameters have a more stable and efficient convergence behavior.

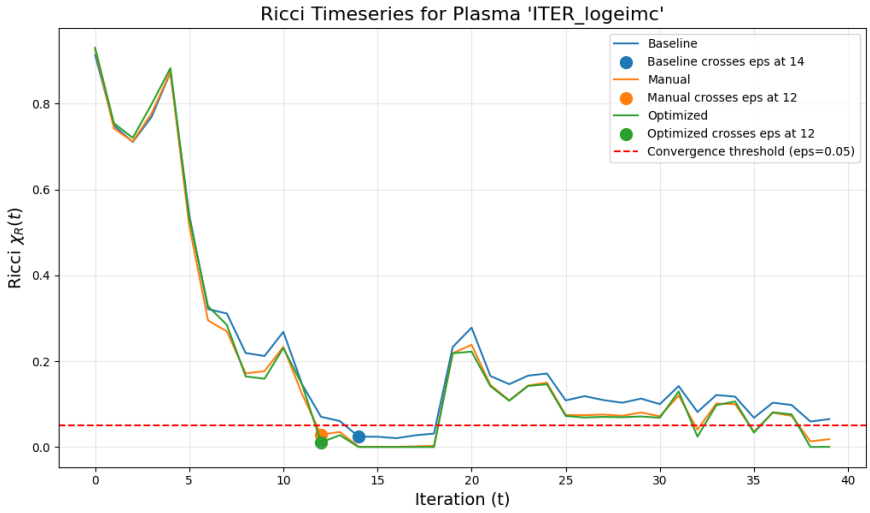

Figure 4.7. Example of Ricci metric evolution $\chi_R(t)$ for one plasma simulation using the baseline, manual, and optimized parameters. The optimized and manual metrics detect convergence earlier than the baseline.

To further assess the reliability of the optimized Ricci metric, Figure 4.8 shows its behavior for a plasma that did not converge (*CMOD_posteriormean*). In this example, none of the configurations reach the convergence threshold $\varepsilon = 0.05$. The optimized configuration does not falsely indicate convergence, this ensures that the improvement of the performance of the parameters does not come at the expense of increased false positives.

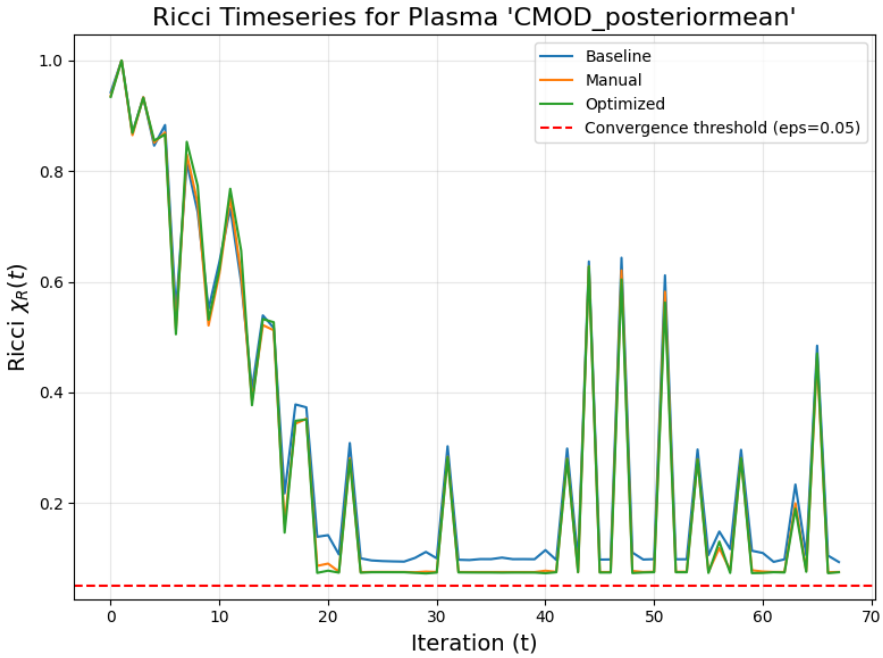

Figure 4.8. Ricci metric evolution $\chi_R(t)$ for a non-convergent plasma (*CMOD_posteriormean*). None of the configurations reach the convergence threshold $\varepsilon = 0.05$, confirming that the optimized Ricci metric does not produce false positives.

These results confirm that the calibration of the Ricci metric enhances its reliability. In convergent cases such as *ITER_logeimc*, the optimized configuration accelerates convergence detection when compared with the baseline. Meanwhile, in non-convergent cases such as *CMOD_posteriormean*, the Ricci score correctly remains above the threshold $\varepsilon = 0.05$ for all configurations, indicating that the optimization process did not introduce false positives. This balance between sensitivity and robustness demonstrates that the calibrated Ricci metric generalizes well across diverse plasma conditions while maintaining physical consistency.

5. Discussion

5.1. Interpretation of results

The results provided in Chapter 4 confirm that the calibration of the Ricci metric presents an improvement in convergence detection within the PORTALS framework. In all the experiments, the optimized configuration consistently reduced the average penalized RMSE with respect to both the default and manually tuned parameters.

In practically all of the experiments, the GA converged toward a narrow region of the parameter space, approximately $d_0 \in [1.95, 2.00]$ and $\lambda \in [0.15, 0.25]$ which indicates that the fitness value has a single dominant minimum rather than multiple optima. This supports the idea that the Ricci metric has a defined optimal regime that generalizes reasonably well across different plasmas. The reproducibility of this result across independent runs and folds highlights the internal consistency of the calibration approach.

However, despite this overall consistency, not all folds behaved equally. During the cross-validation process, the optimized metric reached an error of 2.9 ± 3.1, only one of the folds (Fold 0) showed outlier behavior. This may be due to the variability of the plasmas of the subset used for training or testing rather than a problem with the optimization itself.

The optimized parameters ($d_0 = 1.97$, $\lambda = 0.17$) showed great improvements when compared with the baseline ($d_0 = 2.0$, $\lambda = 1.0$) and manual ($d_0 = 2.0$, $\lambda = 0.5$) configuration, by reducing the mean RMSE by approximately 74% and 44% respectively. Moreover, these improvements were achieved without compromising robustness, this means that the metric did not detect convergence for non-convergent plasmas. This balance between faster convergence detection and absence of false positives is a great indicator that the optimized metric behaves in a physically meaningful way.

The analysis of the time-series showed that the optimized parameters detect convergence earlier and produces slightly smoother Ricci trajectories, with reduced oscillations near the threshold. This implies that the metric provides a more stable and interpretable stopping criterion for the simulations, improving the reproducibility of convergence detection across different devices and

acquisition functions.

The findings of this project suggest that the optimization of the parameters for the Ricci metric constitutes a significant step toward a more accurate convergence detection.

5.2. Limitations

Although the results are promising, it is important to recognize some limitations of the project. Firstly, the dataset of labeled simulations used for training and testing was relatively small and relied on manual inspection to determine convergence. This could limit how well the findings apply to other plasma scenarios or acquisition functions not included in the dataset. Additionally, because convergence was determined through subjective visual assessment, there's a risk of bias [35]. Quantitative validation metrics are typically preferred to ensure reproducibility and reduce human bias during optimization.

Additionally, the threshold value of $\varepsilon = 0.05$ for the Ricci metric was chosen based on empirical observations and, while it worked well in many cases, may not be ideal for every situation.

Regarding the methodology, the GA used for the project is fairly sensitive to data partitioning as proved in Section 4.4 by the presence of outliers with high variability and fitness values.

Lastly, the entire optimization and validation process was performed using PORTALS simulations rather than experimental data, meaning that true performance of the calibrated Ricci metric in real world conditions has not been validated whatsoever. Differences in turbulence models, noise levels, or numerical stability could influence the behavior of the metric when applied beyond the current dataset.

5.3. Future work

One of the main conclusions drawn from the experimentation is that the robustness of the GA leaves a lot to be desired, and so, several directions could be explored in order to reduce the dependence on training data of the algorithm. Adding data from different tokamaks and configurations could help the algorithm to generalize better. In addition to this, automating the labeling process could not only speed it up but also would reduce the human bias introduced by

the current method of visual inspection. Using statistical criteria extracted from the simulation data (e.g., variance stabilization or uncertainty thresholds) could make the dataset more scalable and reproducible.

Regarding the optimization method, some alternative algorithms like Particle Swarm Optimization, Bayesian Optimization or even a multi-objective evolutionary algorithm could be investigated to further explore the optimal Ricci parameters, to try to balance sensitivity and robustness more effectively.

Another possible improvement that was briefly tested in this work is the introduction of the threshold for the Ricci metric as a tunable variable for the optimization instead of using a fixed $\varepsilon = 0.05$. Preliminary experiments showed that by using the right fitness function and heavily punishing the algorithm if there is a false positive, one can make sure the threshold does not default to the highest and it can indeed be representative of the underlying physics.

Finally, another interesting experiment would be to train a machine learning model to predict the optimal (d_0, λ) values for each plasma simulation. This model could then be compared against the parameters obtained through the GA. Although it is expected to perform better by adapting to each individual case, the main point of the experiment would be to evaluate how much improvement it actually provides compared to the GA.

6. Social and Environmental Impact

6.1. Environmental Impact

The project has a positive impact on the environment as it contributes to nuclear fusion research, a clean energy source that emits no greenhouse gases and generates no long-lived radioactive waste. Fusion represents one of the most promising alternatives to tackle future energy and climate challenges, offering a continuous, safe, and scalable supply compared to current options.

Plasma simulations are essential for the design of tokamaks, which is why tools like PORTALS play a key role in validating and optimizing transport models. By improving the efficiency of these tools, in this case, through the optimization of the Ricci metric parameters d_0 and λ, the plasma simulation process becomes significantly more computationally efficient. As a result, more experiments can be conducted, accelerating the progress of nuclear fusion research.

6.2. Social Impact

The project has an overall positive societal impact by contributing to the development of a new technology that can provide clean and sustainable energy, which can improve the quality of life in regions with limited energy access, reducing inequalities and promoting economic and social development.

Although this work does not directly involve human interaction or the use of personal data, its contribution to improving simulation tools for fusion can have significant indirect effects. By accelerating the research and optimization of fusion reactors, the project supports progress toward a more equitable, resilient, and globally distributed energy model.

Referencias

[1] T. F. Neiser, F. Jenko, T. A. Carter, *et al.*, "Gyrokinetic gene simulations of diii-d near-edge l-mode plasmas," *Physics of Plasmas*, vol. 26, no. 9, p. 092 510, Sep. 2019, ISSN: 1070-664X. DOI: 10.1063/1.5052047. eprint: https://pubs.aip.org/aip/pop/article-pdf/doi/10.1063/1.5052047/14852930/092510_1_online.pdf. [Online]. Available: https://doi.org/10.1063/1.5052047.

[2] J. Kates-Harbeck, A. Svyatkovskiy, and W. Tang, "Predicting disruptive instabilities in controlled fusion plasmas through deep learning," *Nature*, vol. 568, no. 7753, pp. 526–531, 2019. DOI: 10.1038/s41586-019-1116-4. [Online]. Available: https://doi.org/10.1038/s41586-019-1116-4.

[3] L. Spangher, A. M. Wang, A. Maris, *et al.*, "Position: Opportunities exist for machine learning in magnetic fusion energy," in *Forty-first International Conference on Machine Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=arwP5FA2dO.

[4] P. Rodriguez-Fernandez, N. T. Howard, A. Saltzman, *et al.*, *Enhancing predictive capabilities in fusion burning plasmas through surrogate-based optimization in core transport solvers*, 2024. arXiv: 2312.12610 [physics.plasm-ph]. [Online]. Available: https://arxiv.org/abs/2312.12610.

[5] P. Ricci, C. Theiler, A. Fasoli, *et al.*, *Physics of Plasmas*, vol. 18, no. 3, p. 032 109, Mar. 2011, ISSN: 1070-664X. DOI: 10.1063/1.3559436. eprint: https://pubs.aip.org/aip/pop/article-pdf/doi/10.1063/1.3559436/16020374/032109_1_online.pdf. [Online]. Available: https://doi.org/10.1063/1.3559436.

[6] E. Institute and O. W. in Data, *Statistical review of world energy (2025) – with major processing by our world in data. "other renewables (including geothermal and biomass)"*, Original data from the Energy Institute's *Statistical Review of World Energy* (2025), processed and visualized by Our World in Data, Energy Institute, 2025. [Online]. Available: https://ourworldindata.org/energy-mix.

[7] IPCC, *Climate change 2023: Synthesis report. contribution of working groups i, ii and iii to the sixth assessment report of the intergovernmental panel on climate change*, DOI:

10.59327/IPCC/AR6-9789291691647, Geneva, Switzerland, 2023. [Online]. Available: https://www.ipcc.ch/report/sixth-assessment-report-cycle/.

[8] H. Ritchie, P. Rosado, and M. Roser, "Co_2 and greenhouse gas emissions," *Our World in Data*, 2023, https://ourworldindata.org/co2-and-greenhouse-gas-emissions.

[9] H. Abu-Shawareb, R. Acree, P. Adams, *et al.*, "Achievement of target gain larger than unity in an inertial fusion experiment," *Phys. Rev. Lett.*, vol. 132, p. 065 102, 6 Feb. 2024. DOI: 10.1103/PhysRevLett.132.065102. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.132.065102.

[10] B. D. Bondarenko, "Role played by o a lavrent'ev in the formulation of the problem and the initiation of research into controlled nuclear fusion in the ussr," *Phys. Usp.*, vol. 44, no. 8, pp. 844–851, 2001. DOI: 10.1070/PU2001v044n08ABEH000910. [Online]. Available: https://ufn.ru/en/articles/2001/8/r/.

[11] I. Organization. "In a few lines." Accessed: 2025-08-29. (n.d.), [Online]. Available: https://www.iter.org/few-lines.

[12] J. D. Lawson, "Some criteria for a power producing thermonuclear reactor," *Proceedings of the Physical Society. Section B*, vol. 70, no. 1, p. 6, Jan. 1957. DOI: 10.1088/0370-1301/70/1/303. [Online]. Available: https://dx.doi.org/10.1088/0370-1301/70/1/303.

[13] A. J. Creely, M. J. Greenwald, S. B. Ballinger, *et al.*, "Overview of the sparc tokamak," *Journal of Plasma Physics*, vol. 86, no. 5, p. 865 860 502, 2020. DOI: 10.1017/S0022377820001257.

[14] *Diii-d national fusion facility*, https://d3dfusion.org/, n.d.

[15] J. Candy and R. E. Waltz, "Anomalous transport scaling in the diii-d tokamak matched by supercomputer simulation," *Phys. Rev. Lett.*, vol. 91, p. 045 001, 4 Jul. 2003. DOI: 10.1103/PhysRevLett.91.045001. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.91.045001.

[16] A. Pankin, J. Breslau, M. Gorelenkova, *et al.*, "Transp integrated modeling code for interpretive and predictive analysis of tokamak plasmas," *Computer Physics Communications*, vol. 312, p. 109 611, 2025, ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2025.109611. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010465525001134.

[17] G. Pereverzev, P. Yushmanov, A. Dnestrovskii, *et al.*, *ASTRA - an automatic system for transport analysis in a tokamak*, Jan. 2025.

[18] G. M. Staebler, J. E. Kinsey, and R. E. Waltz, "Gyro-landau fluid equations for trapped and passing particles," *Physics of Plasmas*, vol. 12, no. 10, p. 102 508, Oct. 2005, ISSN: 1070-664X. DOI: 10.1063/1.2044587. eprint: https://pubs.aip.org/aip/pop/article-pdf/doi/10.1063/1.2044587/14688354/102508_1_online.pdf. [Online]. Available: https://doi.org/10.1063/1.2044587.

[19] J. Candy, E. Belli, and R. Bravenec, "A high-accuracy eulerian gyrokinetic solver for collisional plasmas," *Journal of Computational Physics*, vol. 324, pp. 73–93, 2016, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2016.07.039. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999116303400.

[20] O. Meneghini, S. Smith, L. Lao, *et al.*, "Integrated modeling applications for tokamak experiments with omfit," *Nuclear Fusion*, vol. 55, no. 8, p. 083 008, 2015. [Online]. Available: http://iopscience.iop.org/article/10.1088/0029-5515/55/8/083008/meta.

[21] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, Nov. 2005, ISBN: 9780262256834. DOI: 10.7551/mitpress/3206.001.0001. eprint: https://direct.mit.edu/book-pdf/2514321/book_9780262256834.pdf. [Online]. Available: https://doi.org/10.7551/mitpress/3206.001.0001.

[22] S. Jalas, M. Kirchen, P. Messner, *et al.*, "Bayesian optimization of a laser-plasma accelerator," *Phys. Rev. Lett.*, vol. 126, p. 104 801, 10 Mar. 2021. DOI: 10.1103/PhysRevLett.126.104801. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.126.104801.

[23] R. J. Shalloo, S. J. D. Dann, J. .-. Gruse, *et al.*, "Automation and control of laser wakefield accelerators using bayesian optimization," *Nature Communications*, vol. 11, no. 1, p. 6355, 2020. DOI: 10.1038/s41467-020-20245-6. [Online]. Available: https://doi.org/10.1038/s41467-020-20245-6.

[24] A. Ferran Pousa, S. Jalas, M. Kirchen, *et al.*, "Bayesian optimization of laser-plasma accelerators assisted by reduced physical models," *Phys. Rev. Accel. Beams*, vol. 26, p. 084 601, 8 Aug. 2023. DOI: 10.1103/PhysRevAccelBeams.26.084601. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevAccelBeams.26.084601.

[25] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. The MIT Press, Apr. 1992, ISBN: 9780262275552. DOI: 10.7551/mitpress/1090.001.0001. [Online]. Available: https://doi.org/10.7551/mitpress/1090.001.0001.

[26] K. Deb, "Multiobjective optimization using evolutionary algorithms. wiley, new york," in Jan. 2001.

[27] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," in ser. Foundations of Genetic Algorithms, G. J. RAWLINS, Ed., vol. 1, Elsevier, 1991, pp. 69–93. DOI: https://doi.org/10.1016/B978-0-08-050684-5.50008-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780080506845500082.

[28] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, 1995. [Online]. Available: https://api.semanticscholar.org/CorpusID:18860538.

[29] K. Deb and M. Goyal, "A combined genetic adaptive search (geneas) for engineering design," 1996. [Online]. Available: https://api.semanticscholar.org/CorpusID:18387364.

[30] A. Eiben and J. Smith, *Introduction To Evolutionary Computing*. Jan. 2003, vol. 45, ISBN: 978-3-642-07285-7. DOI: 10.1007/978-3-662-05094-1.

[31] Z. Michalewicz and M. Schoenauer, "Evolutionary Algorithms for Constrained Parameter Optimization Problems," *Evolutionary Computation*, vol. 4, no. 1, pp. 1–32, Mar. 1996. [Online]. Available: https://inria.hal.science/hal-02986407.

[32] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *CoRR*, vol. abs/2002.04504, 2020. arXiv: 2002.04504. [Online]. Available: https://arxiv.org/abs/2002.04504.

[33] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937, ISSN: 01621459, 1537274X. [Online]. Available: http://www.jstor.org/stable/2279372 (visited on 10/12/2025).

[34] P. B. NEMENYI, "Distribution-free multiple comparisons," English, Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2023-02-17, Ph.D. dissertation, 1963, p. 127, ISBN: 9781084733008. [Online]. Available: https://www.proquest.com/dissertations-theses/distribution-free-multiple-comparisons/docview/302256074/se-2.

[35] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade: Second Edition*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 437–478, ISBN: 978-3-642-35289-8. DOI: 10.1007/978-3-642-35289-8_26. [Online]. Available: https://doi.org/10.1007/978-3-642-35289-8_26.

Índice de términos

Acronyms

MIT The Massachusetts Institute of Technology (MIT) is a private research university in Cambridge, Massachusetts, United States. Established in 1861, MIT has played a significant role in the development of many areas of modern technology and science.. 2, 9

NIF National Ignition Facility. 7

PM Polynomial Mutation. 18, 27, 32

PORTALS PORTALS is a surrogate-based framework that accelerates steady-state plasma transport simulations using Bayesian optimization.. iii, 2--5, 12--15, 22, 23, 38, 42, 43, 45

PSFC The Plasma Science and Fusion Center at the Massachusetts Institute of Technology is a university research center for the study of plasmas, fusion science and technology.. 2, 9

PSO Particle Swarm Optimization. 21

RMSE The Root Mean Squared Error (RMSE) measures the average magnitude of the squared differences between predicted and true values, penalizing larger errors more heavily.. iv, v, 21, 27, 33, 36--39, 42

SBX Simulated Binary Crossover. 18, 27, 32

SPARC Smallest Possible ARC. 9

TGLF Trapped Gyro-Landau Fluid. 12