1. Introduction

1.1. Project Overview

The integration of Machine Learning-based techniques in Scientific Computing is nowadays becoming more and more popular. A challenging scenario is the one of uncertainty quantification, where the efficiency of the Neural Networks can be help in overcoming the computational costs of demanding approaches based on PDE solvers.

Bayesian Physics-Informed Neural Networks (B-PINNs) represent a way to address such problems, thanks both to the inclusion of the residuals of differential equations during the learning process and to training algorithms which reconstruct a distribution of the Network's output rather than a single prediction.

This method, recently proposed in few papers such as [11] and [7], is an evolution of Physics-Informed Neural Networks (PINNs), a Deep Learning framework for solving problems involving partial differential equations ([9], [10]); B-PINNs take inspiration from the theory of Bayesian Statistics to empower the PINNs method and enable the introduction of uncertainty quantification.

In this project, we implemented a library for Bayesian Physics-Informed Machine Learning designed to be modular, flexible and open to plugins. In the state-of-the-art of this niche application of PINNs, it is indeed more common to find implementations of a specific Bayesian training method applied to one differential problem; therefore, the goal we set was the development of a library with a wide offer of features.

Given the project aim, the design of some components of the library played a crucial role. The backbone of the method was thought as a rigid modular structure, built in such a way that the blocks for the desired application could be assembled into it without affecting the skeleton. Each block can be defined with its own characteristic features, that are a small number and do not introduce heavy constraints on the functionalities, so that it is easy to generate new blocks for different methods or problems in which we may be interested.

The library's characteristics are then highlighted by a selection of applications, aimed at showing the main pillars: the set-up of four training algorithms very different in structure and intuition one from each other, the contribution of the introduction of the physical information into the model and the library portability to higher dimensions.

For the method performance on the specific applications, the parameters' tuning plays a crucial role. Though in the project we focused more on the horizontal expansion of library feature, we stressed on performance quality for the case of one specific Bayesian training algorithm with sustainable computational time and interesting challenges in parameters' choices, developing a wider variety of test cases with fine-tuned method options.

1.2. Report Structure

The theoretical foundations of the proposed implementation are presented in section 2, with the explanation of the building blocks of a B-PINN. Starting from the basics of neural networks and from deterministic training algorithms described in subsection 2.1, we then illustrate the theory behind Physics-Informed Machine Learning in subsection 2.2. Then, we present the learning mechanisms of Bayesian Neural Networks in subsection 2.3 and illustrate how to combine them with PINNs to obtain B-PINNs in subsection 2.4.

In section 3 we deepen into the non-deterministic training algorithms implemented. We illustrate the theoretical background behind them and their procedures, problematizing and discussing the choice of parameters but without deepening into the technical implementation details (postponed to section 5). Each subsection is dedicated to one algorithm: subsection 3.1 to Hamiltonian Monte Carlo, subsection 3.2 to Variational Inference and subsection 3.3 to Stein Variational Gradient Descent.

In section 4 we first present and motivate the coding choices in terms of environment (subsection 4.1) and libraries (subsection 4.2). Then, we devote subsection 4.3 to Object-Oriented Programming in Python, focusing on the features implemented in the project. Finally, subsection 4.4 presents the structure of the repository and the content of the configuration, data and outputs files.

In section 5, we illustrate the source code: we describe the three executable scripts devote specific sections to the various modules, whose tasks are related to the implementation of the B-PINNs method or to post-processing utilities such as UQ and performance evaluation.

In section 6, we present the application of the library implemented to a series of test cases, with different datasets and training configurations. The showcase of applications is organized in such a way to highlight separately the main features of the library: in subsection 6.1, we focus on a comparison of the training algorithms on the same task. Then, in subsection 6.2, we show the power of the inclusion of the physical information on the Damped Harmonic Oscillator problem. In subsection 6.3 we exhibit an example in 2D dimension and finally in subsection 6.4 we present the results on several applications obtained with a massive fine-tuning of a model trained with HMC.

Finally, in section 7, we draw conclusions on this project and suggest further developments.