Differentiable Simulators for HEP

L. Heinrich[1], M. Kagan*[2], M. Mooney[3], and K. Terao[2]

[1]CERN
[2]SLAC National Accelerator Laboratory
[3]Colorado State University

Thematic Areas: (check all that apply □/■)
□ (CompF1) Experimental Algorithm Parallelization
■ (CompF2) Theoretical Calculations and Simulation
■ (CompF3) Machine Learning
□ (CompF4) Storage and processing resource access (Facility and Infrastructure R&D)
□ (CompF5) End user analysis
□ (CompF6) Quantum computing
□ (CompF7) Reinterpretation and long-term preservation of data and code
□ (Other) *[Please specify frontier/topical group]*

*Corresponding Author: Michael Kagan (makagan@slac.stanford.edu)

Introduction

Scientific simulators serve as a keystone for encapsulating knowledge acquired within scientific domains. Simulators typically depend on parameters of an underlying theory or model and provide a means to generate sample observation from a stochastic process involving latent or unobserved random variables. However, the density of generated observations is not known and in many cases is analytically intractable. Inference of model parameters and latent variables using simulators is a major challenge, and techniques such as likelihood-free [1] and variational inference [2] approach this challenge through learning an appropriate ML model from large simulated datasets. Information may be lost in this process. The goal of the research direction discussed in this LOI is to investigate a different approach to utilizing simulators by enabling the use of exact differentiable simulators, e.g. simulators interfaced with automatic differentiation frameworks [3], within ML and optimization pipelines for scientific data analysis. As gradient descent drives a vast array of algorithms for optimization, training, and inference, utilizing a simulator within such an ML pipeline requires it to produce differentiable predictions in order to propagate gradients to other parts of the ML pipeline. In this way, a differentiable simulator may act as a complex and dynamic reparameterization of its inputs that can be used seamlessly with ML tools to produce powerful differentiable programs.

Interfacing with Automatic Differentiation

Differentiable simulators can be built using automatic differentiation (AD) frameworks [3]. AD represents fundamental arithmetic operations and elementary functions within a computation graph, and applies the chain rule to automatically compute high precision derivatives of any order. Several frameworks exist to create or augment programs to enable AD in a variety of computing languages. As modern deep learning frameworks are built upon AD, it will be important to examine the AD implementation of simulators within these frameworks to facilitate their use in harmony with ML algorithms. For simulators, a variety of considerations for the AD implementation and interaction with ML tools should be studied, including interfacing or porting simulation code to the ML framework language, dealing with the dynamic control flow of simulators within AD computational graphs, and reducing additional latency incurred by ML frameworks that are optimized for performing common ML operations but potentially not for the simulators. More generally, it will be vital to understand how to control computational costs in order to scale up the production and utilization for the large simulated datasets in HEP applications.

Inference with Differentiable Simulators

From the standpoint of statistical modeling and ML, differentiable simulators serve as flexible and precise models that map parameters and stochastic inputs to an experiment's observation space. The samples produced by these simulators are differentiable and well suited for gradient-based optimization and learning. This research direction should explore how ML algorithms can take advantage of the simulator differentiability. For example, experimental design optimization and parameter tuning, differentiable simulators could enable the use of gradient descent for MLE and MAP estimation, and adversarial optimization [4]. For parameter inference, differentiable simulators could serve as a likelihood or decoder in variational inference, adversarially learned inference [5], neural density estimation [6], or Hamiltonian Monte Carlo [7]. More generally, the potential for differentiable simulators to improve the precision of inference strategies in inverse problems should be explored.

Implementations in HEP Frontiers

The aim of this research direction is to explore the implementation and use of differentiable state-of-the-art simulators for High Energy Physics applications. Examples of differentiable HEP simulators that have begun to be explored include: (i) Simulation of matrix element scattering amplitudes for colliders. Such simulators enable the evaluation and sampling of particle configurations occurring in particle collisions. These differentiable simulators could then applied in: collision event reconstruction, parameter inference, and ML-based density estimation. (ii) Simulation of neutrino-nucleus/many-body intranuclear interaction modeling and the response of liquid argon time projection chambers, for which analysis techniques are still emergent despite their wide use in neutrino experiments. Such differentiable simulators can be applied in: simulator tuning to match observed data and forward modeling to infer the impact of parameter distributions to the physics output.

Summary

This LOI considers the investigation of differentiable scientific simulators for high energy physics, as well as how to captialize on the capabilities of differentiable simulators in concert with ML to tackle challenges in scientific data analysis such as tuning, optimization, and inference. This research direction requires investigating how to augment scientific simulators within automatic differentiation frameworks [3], and how to exploit the differentiable capability of these enhanced simulators within ML pipelines.

References

[1] Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 2020.

[2] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(4):1303–1347, 2013.

[3] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18:1–43, 2018.

[4] Gilles Louppe, Joeri Hermans, and Kyle Cranmer. Adversarial variational optimization of non-differentiable simulators. *PMLR*, 89:1438–1447, 2019.

[5] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv:1606.00704*, 2016.

[6] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *arXiv:1912.02762*, 2019.

[7] Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216 – 222, 1987.