HEColliderBit: A GAMBIT module for the calculation of high energy collider observables

The GAMBIT Collaboration: Alphabetical author list

January 5, 2015

Abstract

We present a new code for the calculation of high energy collider observables, given a generic theory of beyond the Standard Model physics. Describe novel features of HEColliderBit (parallelised MC generation, fast SUSY cross-section calculations, generic interface to BSM models, lots of LHC analyses). Mention link to GAMBIT framework, but emphasise that the package presents a standalone solution to the problem of applying LHC constraints to new physics theories.

1 Introduction

Wealth of recent LHC data provides strong constraints on new physics models. It remains difficult to apply LHC constraints to generic physics models in a rigorous fashion. Various attempts have been made to automate the process (SmodelS, CheckMATE, etc). No attempt has attacked the problem in a fully rigorous way, and neither have any existing attempts been integrated into a fully general framework for statistical fits of generic BSM theories.

We present a new, fully general solution based on parallelised Monte Carlo simulation, fast detector simulation and fast cross-section calculations for SUSY processes. Part of the Global and Module BSM Inference Tool, which will include, etc.

This paper is organised as follows. In Section ?? we..., etc.

2 Physics background

Describe the basic design elements of the package, i.e. what we calculate given a model of BSM physics, but with the full details of the calculation

deferred to the following section. Need to emphasise that the user may pick and choose each element of the calculation.

- Number of signal events for cut and count analyses (or in a particular bin of a binned likelihood fit) is $\sigma \times A \times \epsilon \times \mathcal{L}$.
- State very briefly how each element of the calculation above is performed in HEColliderBit.
- Define likelihoods used in LHC searches based on the signal yields.

3 Code description

For each part of the calculation outlined above: describe the code that exists within GAMBIT and/or the ready-made interface to a standard HEP code. Show validation plots (with possible links to appendices).

3.1 User interface

- Describe interface to GAMBIT framework.
- Describe standalone interface (how to configure and run the package, what the defaults are).
- Describe the interface to the models that we include and how to change the input parameters of those models.
- Explain how to interface with a new model.
- Explain how to run code in single core or multicore form (i.e using the standard OpenMP commands).
- Describe parameters that can be varied in yaml file.

3.2 Cross-section calculations

- Describe fast SUSY cross-section calculator design and implementation
- Link to paper if we have an external paper
- Include validation plots
- Describe parameters that can be varied in yaml file.
- Describe how we treat cross-sections for generic models.

3.3 Monte Carlo event generation

- Describe existing Pythia 8 interface (and OpenMP parallelisation)
- Explain how to interface a different MC generator?
- Describe parameters that can be varied in yaml file for Pythia.

3.4 Detector simulation

- Describe DELPHES interface for ATLAS and CMS processes
- FastSim: Give brief description of simulation, with link to FastSim paper?)
- Show some FastSim validation plots? (not really necessary if we have a FastSim paper, but might be useful to show DELPHES vs FastSim plots with speed comparisons)
- Describe parameters that can be varied in yaml file.

3.5 Event analysis framework

- Describe event analysis framework
- Emphasise applicability to simulated and/or reconstructed events
- Describe the LHC analyses included in the module and include some validation plots showing some nicely reproduced ATLAS and CMS limits.
- Explain how to add new analyses

3.6 Statistics

- Describe the various likelihoods one may calculate given a signal yield (overlaps with main GAMBIT framework- perhaps we need to release a small stats offshoot within HEColliderBit for standlone users?))
- Describe parameters that can be varied in yaml file.

4 Examples

Give some tutorial-style examples for common use cases.

4.1 MSSM example

- Reproduction of an LHC CMSSM exclusion limit?
- How to scan over pMSSM parameters and return a likelihood.

4.2 Generic Pythia model example

Give an example of Madgraph matrix element code to Pythia 8, show how to get it working and how to scan over the parameters and return a likelihood. Might be ambitious for the first paper, but it wouldn't take too long and would be a superb feature for increasing the user base.

5 Conclusions

6 Acknowledgements

7 Appendix: Validation (optional)